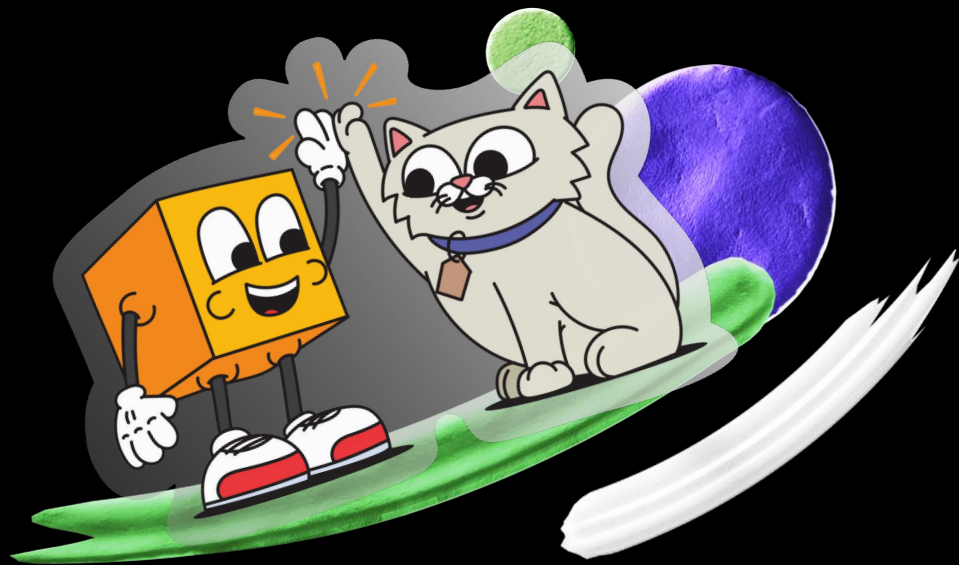


Observability микрофронтендов

Дарья Саенко
Frontend Engineer

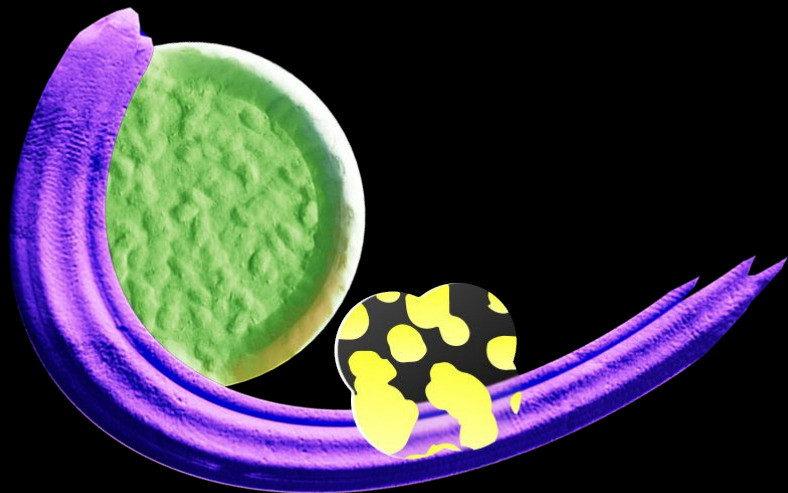


Спикер

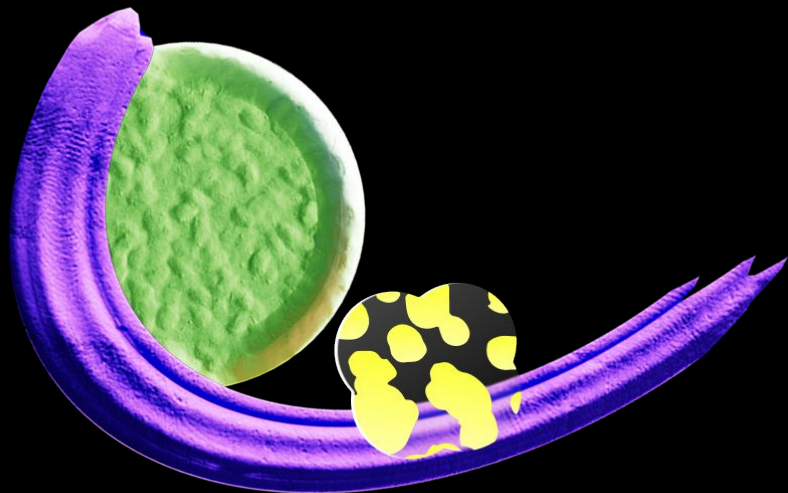


Дарья Саенко Frontend Engineer

Работаю в Авито, в юните Frontend Architecture, разрабатываю решения для микрофронтендов и рантайма.



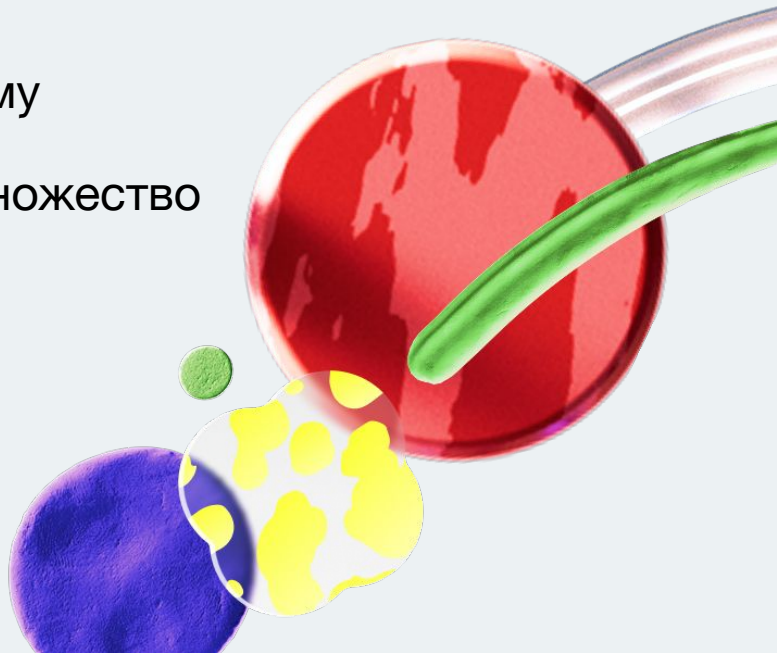
Чем занимается юнит Frontend Architecture в АВИТО



- Проектируем архитектуру фронтенда для упрощения продуктовой разработки
- Разрабатываем инструменты DX и CI, подходы и инструменты тестирования
- Развиваем UI-kit и отвечаем за консистентность всего UI/UX продукта Авито

О чём расскажу в докладе

- ✔ Из чего состоит наша **система observability**
- ✔ Какие инструменты используем
- ✔ Что логируем, какие метрики пишем и почему
- ✔ Как распространяем лучшие практики на множество микрофронтендов



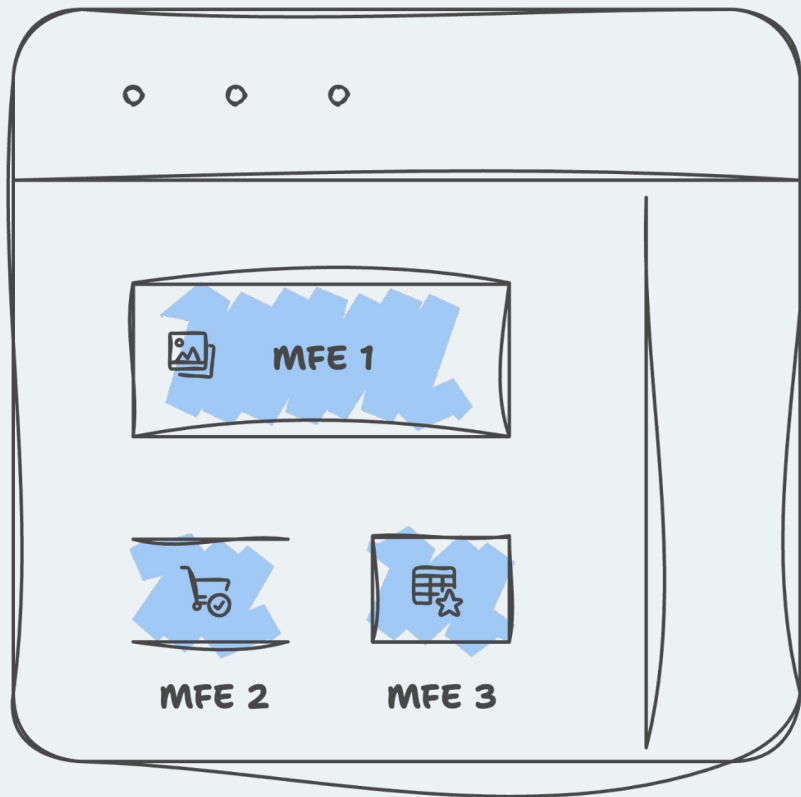
Что такое микрофронтенды



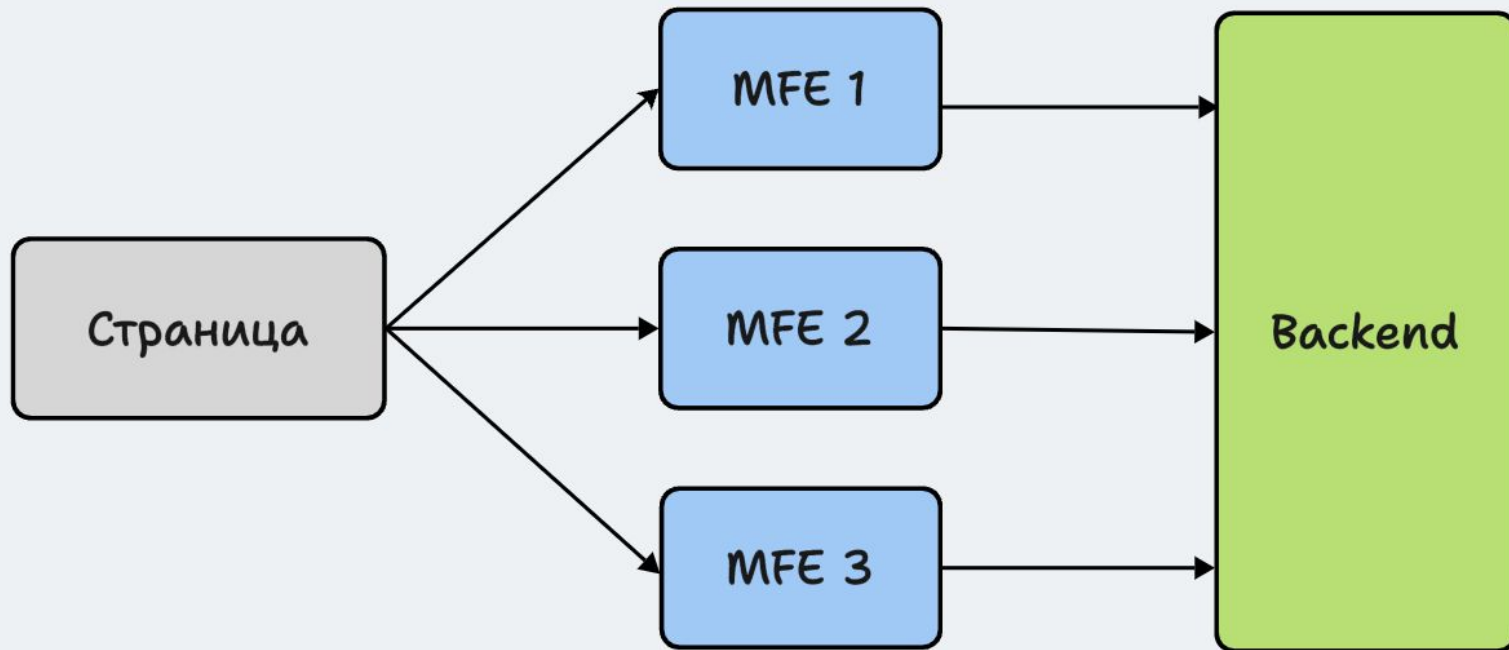
Микрофронтенды

Это архитектурный подход, при котором веб-приложение разбивается на независимые части.

Эти части обычно разрабатываются разными командами и релизятся отдельно друг от друга.



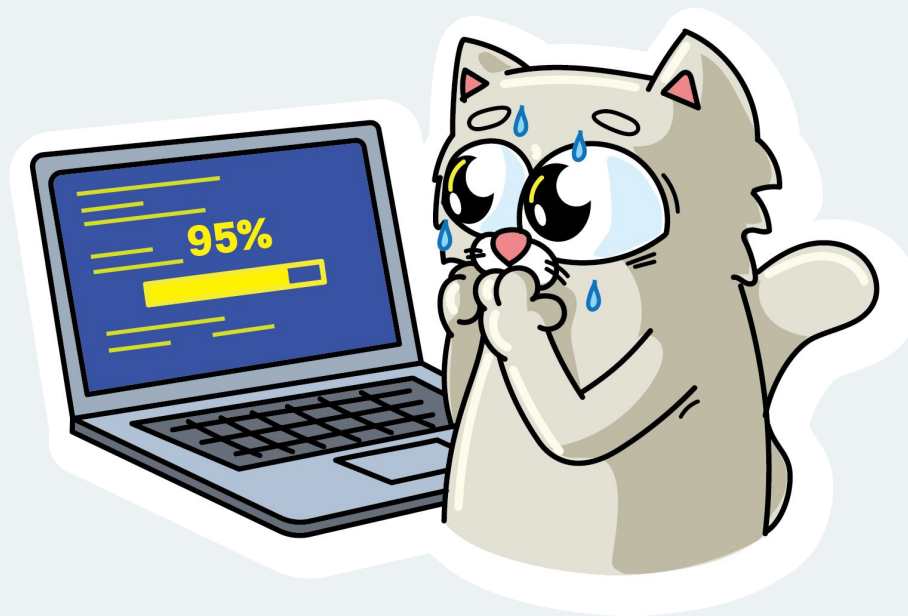
Архитектура микрофронтендов



Зачем нам микрофронтенды

До микрофронтендов у нас были единые **монолит-приложения**, но они имели ряд минусов:

- ⊖ долгий CI/CD;
- ⊖ низкая отказоустойчивость;
- ⊖ сложность детекции проблем;
- ⊖ сложность поддержки.



Микрофронтенды

Снизили TTM

в 5 раз

сократили время деплоя

**за счет деплоя меньшего
объема кода.**

в 10 раз

выросло число релизов

**за счет релизов по требованию
вместо релизов по
расписанию.**

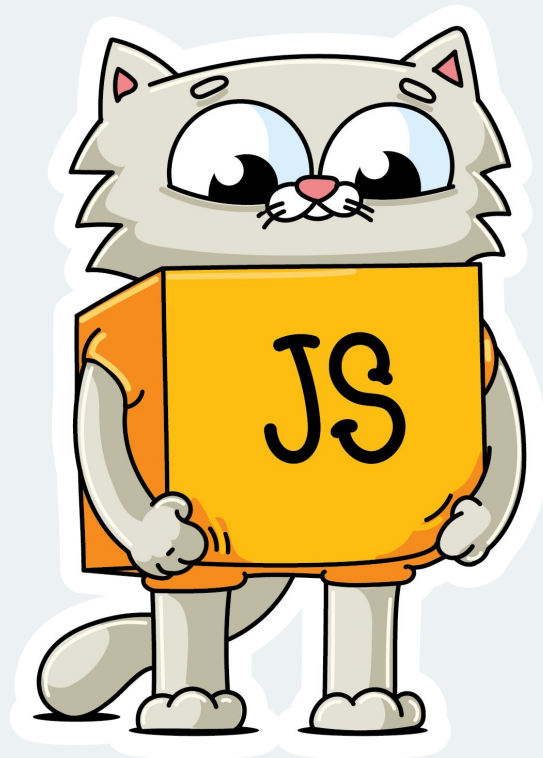
~в 7 раз

сократили число E2E

запускаемых на коммите.

Микрофронтенды позволили

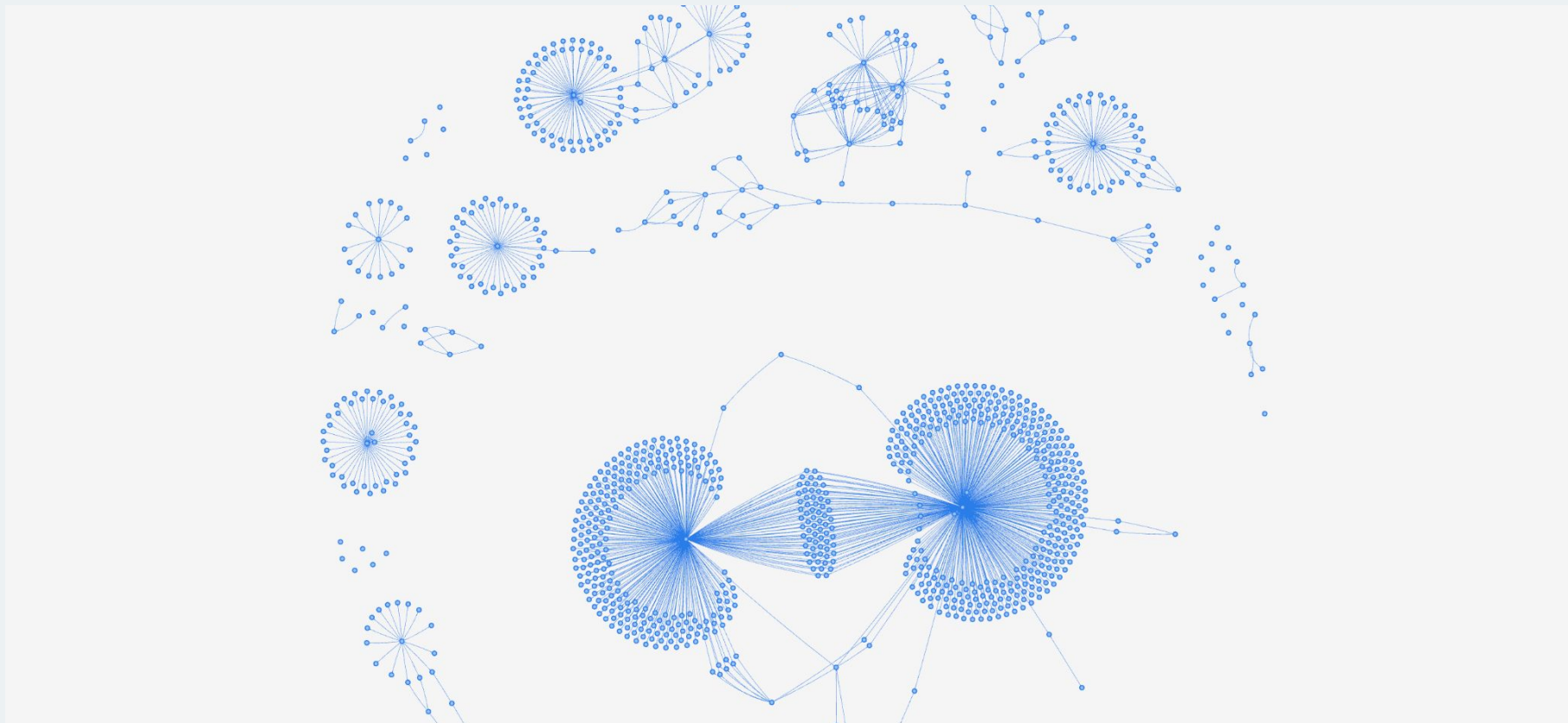
- ✓ **Повысить надежность сайта**
за счет уменьшения связанности с основным приложением и контрактной системе взаимодействия.
- ✓ **Повысить скорость детекции**
за счет меньшего количества задач в релизах и чёткого разделения ответственности.
- ✓ **Уменьшить связанность команд**
за счёт ограничения доступа к внешним ресурсам.



Микрофронтенды

Как это устроено у нас

- ✔ Микрофронтенды **работают через внутреннее решение** на базе Module Federation.
- ✔ Микрофронтенд принадлежит сервису на NodeJs; **сервис может иметь один и более микрофронтендов.**
- ✔ Микрофронтенды рендерятся **на сервере и клиенте.**
- ✔ Микрофронтенды **могут встраиваться** в разные приложения (хосты).



В Авито ~1000 микрофронтендов

Как следить за тысячей микрофронтендов?



Что такое observability



Observability

Это подход, позволяющий в реальном времени получать представление о состоянии системы, чтобы оперативно находить, анализировать и устранять проблемы.



Как достичь надежности

01

**Предотвращать
проблемы**

02

**Реагировать
на проблемы**

03

**Измерять
успех**

Как достичь надежности

01

**Предотвращать
проблемы**

Канареечные релизы

02

**Реагировать
на проблемы**

Алерты

03

**Измерять
успех**

NFR

Как достичь надежности

Термины



Канаречный релиз – метод развертывания сервиса, при котором новая версия сначала становится доступной небольшой группе пользователей.



Алерты – автоматические уведомления о критических событиях и изменениях показателей, требующие внимания разработчика.



NFR – нефункциональные требования, относящиеся прямо или косвенно к производительности сервиса.

Как предотвращаем инциденты

Канареечные релизы

Деплоим **критичные сервисы** через канареечные релизы с автоматическим увеличением процента и автоматическим откатом в случае проблем на основе проберов.



Как предотвращаем инциденты

Канареечные релизы

Пробер – проверка параметра, которая периодически запускается во время деплоя и проверяет установленные границы.

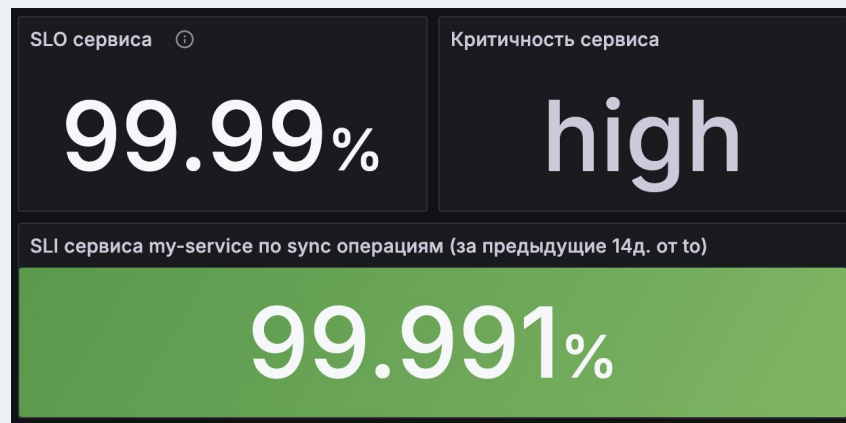
- ✓ autosanary увеличил трафик до 20% ●
- > ✓ Пробер Общее число 500 ошибок по всем страницам 📈 📉 : метрика 2 (ok >=0.00)
- > ✓ Пробер Новые события в Sentry с привязкой к текущему релизу 📈 📉 : метрика 0 (ok >=0.00)
- > ✓ Пробер OS Memory Heap 📈 📉 : метрика 0.96 (ok >=0.00)
- > ✓ Пробер Ошибки рендера на сервере 📈 📉 : метрика 1.1 (ok >=0.00)

Как измеряем успех NFR

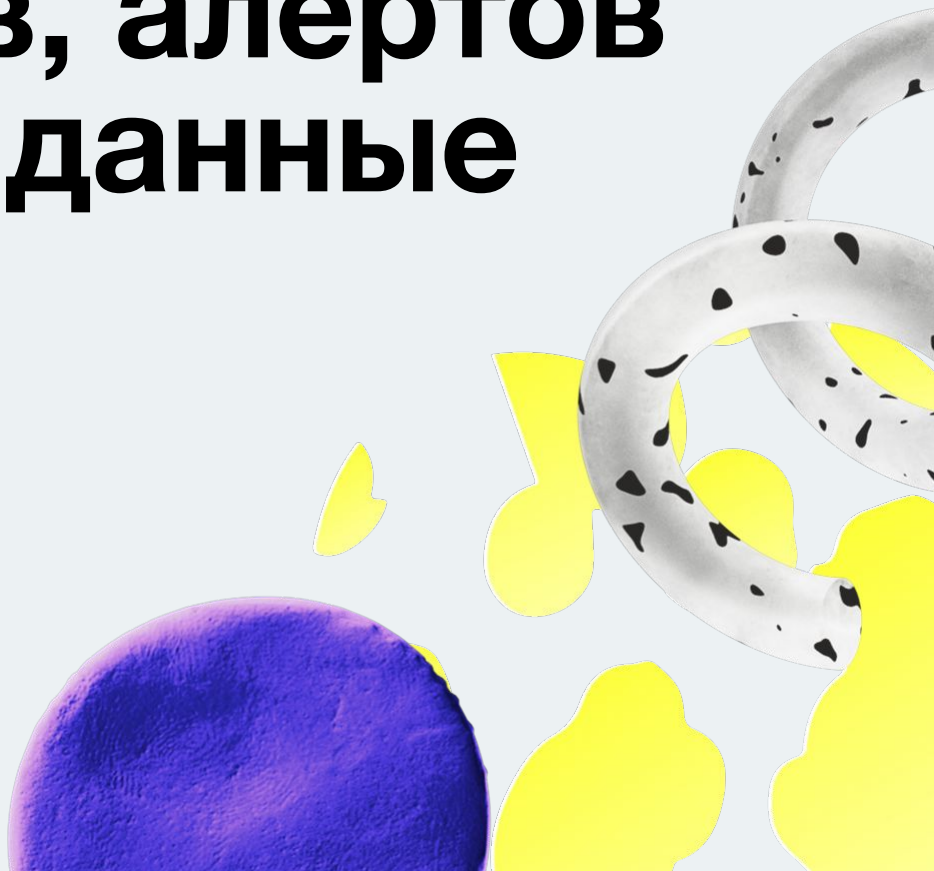
Описываем требования к производительности и надежности, а затем сравниваем с реальными показателями.

SLO – целевой уровень качества.

SLI – текущий уровень качества.



Для проберов, алертов и NFR нужны данные



Из чего состоит observability

01

Логи

Текстовые события,
обычно с контекстом.

02

Метрики

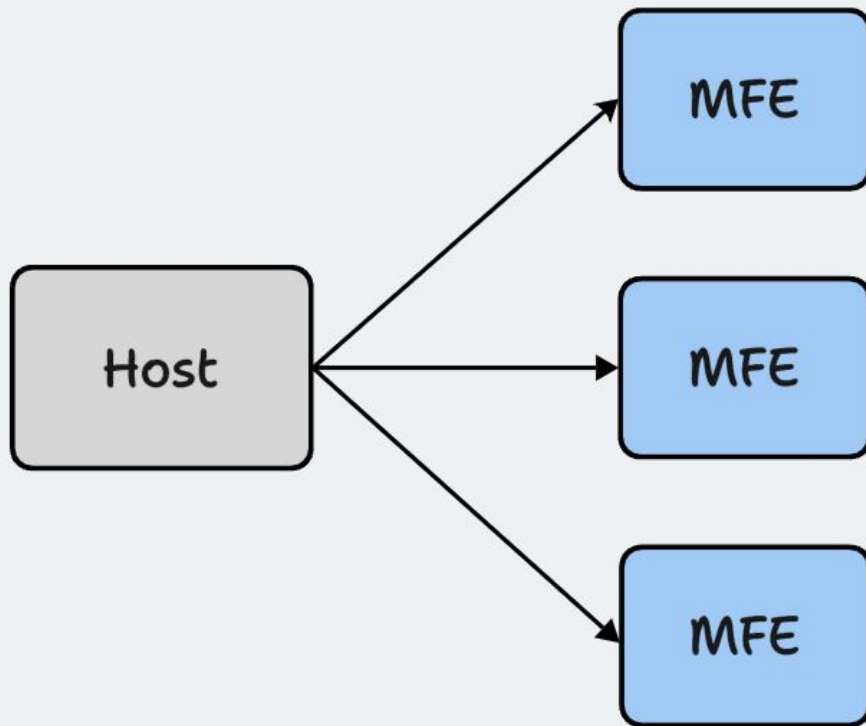
Численные значения,
которые можно
агрегировать
и отображать.

03

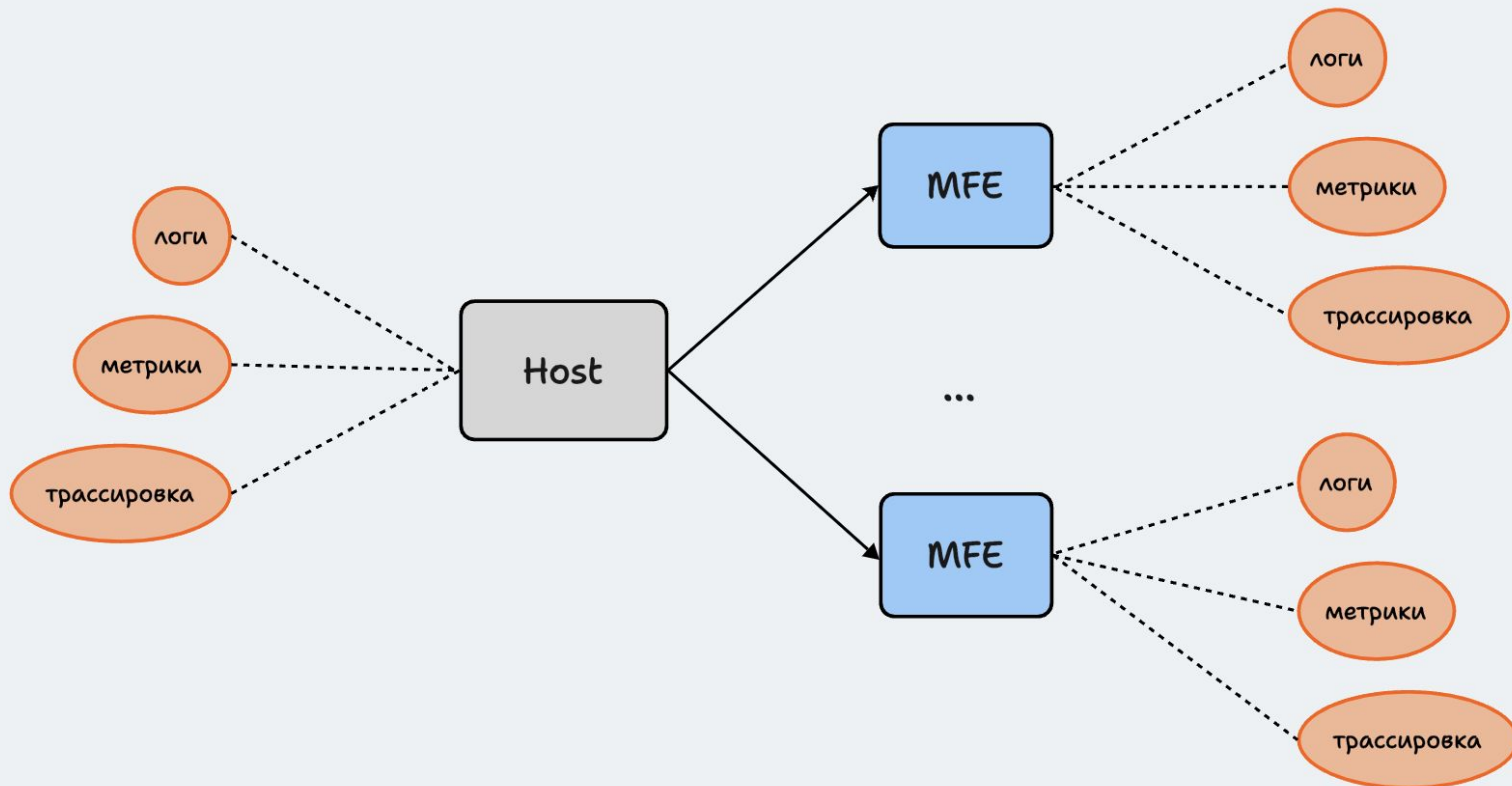
Трассировка

Хронология вызовов
между компонентами
системы.

Архитектура микрофронтендов



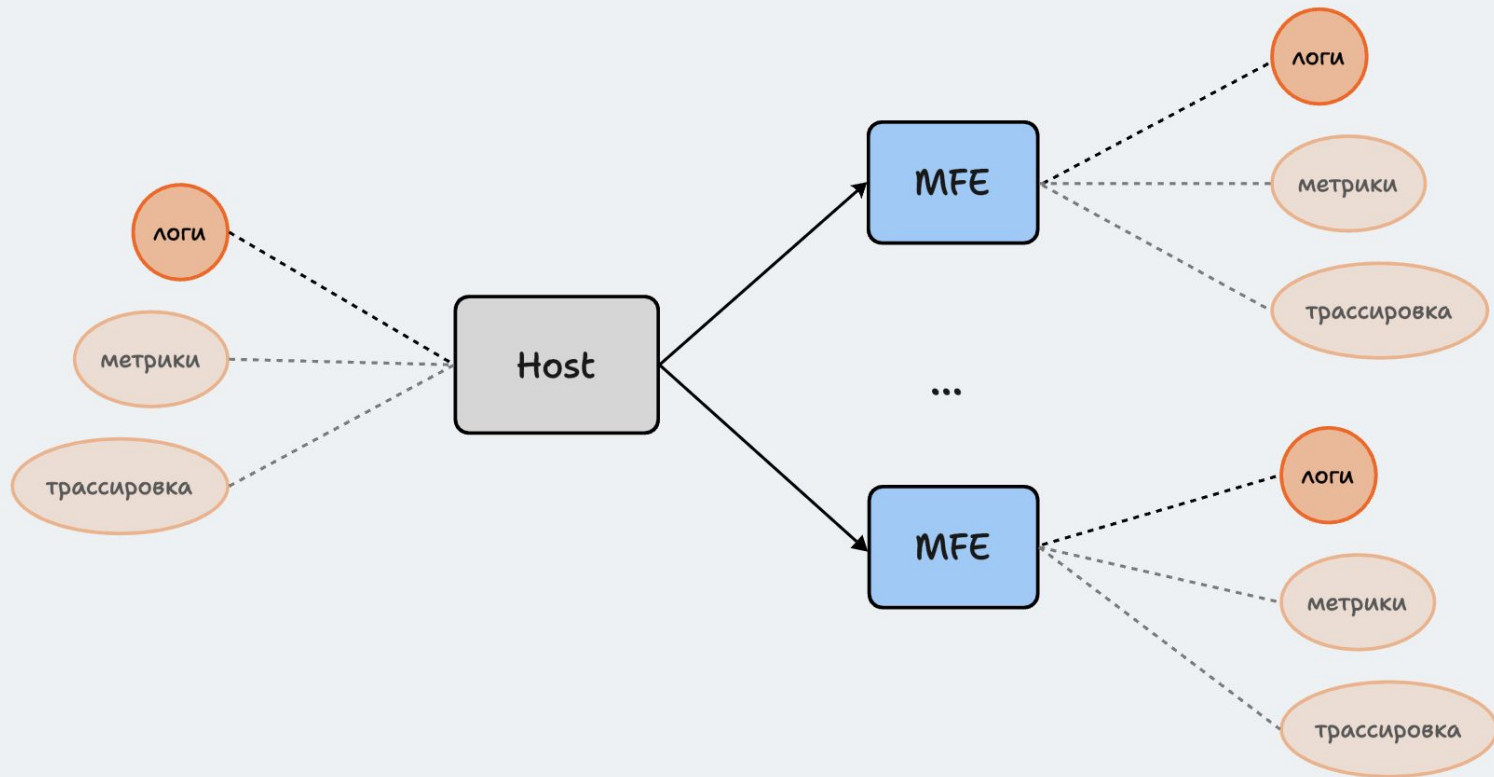
Observability на схеме



Observability. Логи



Логи

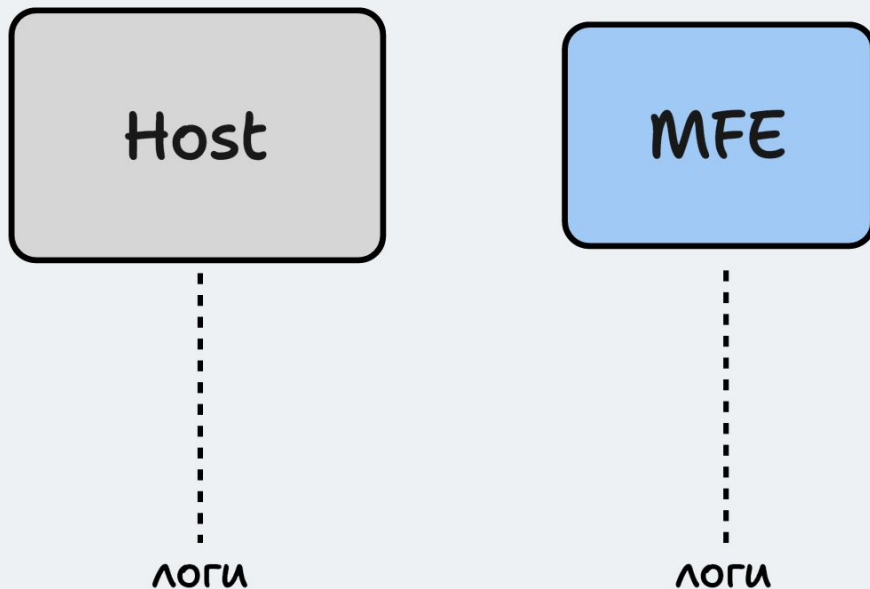


Логи

Откуда писать

Логи микрофронтенда пишем на сервере и клиенте из сервиса, в котором находится микрофронтенд.

Хост пишет только собственные логи.



Логи

Откуда писать – альтернативный вариант

Если писать логи о микрофронте **из хоста**:

- ⊖ система будет менее надежной;
- ⊖ небезопасно;
- ⊖ нет гарантии, что хост собирает логи.

Если что-то сломается в хосте, не будет информации о его микрофронтендах

Есть риск потерять логи

Логи

Что писать

01 Ошибки

Но исключаем те, которые невозможно исправить.

02 Данные для отладки

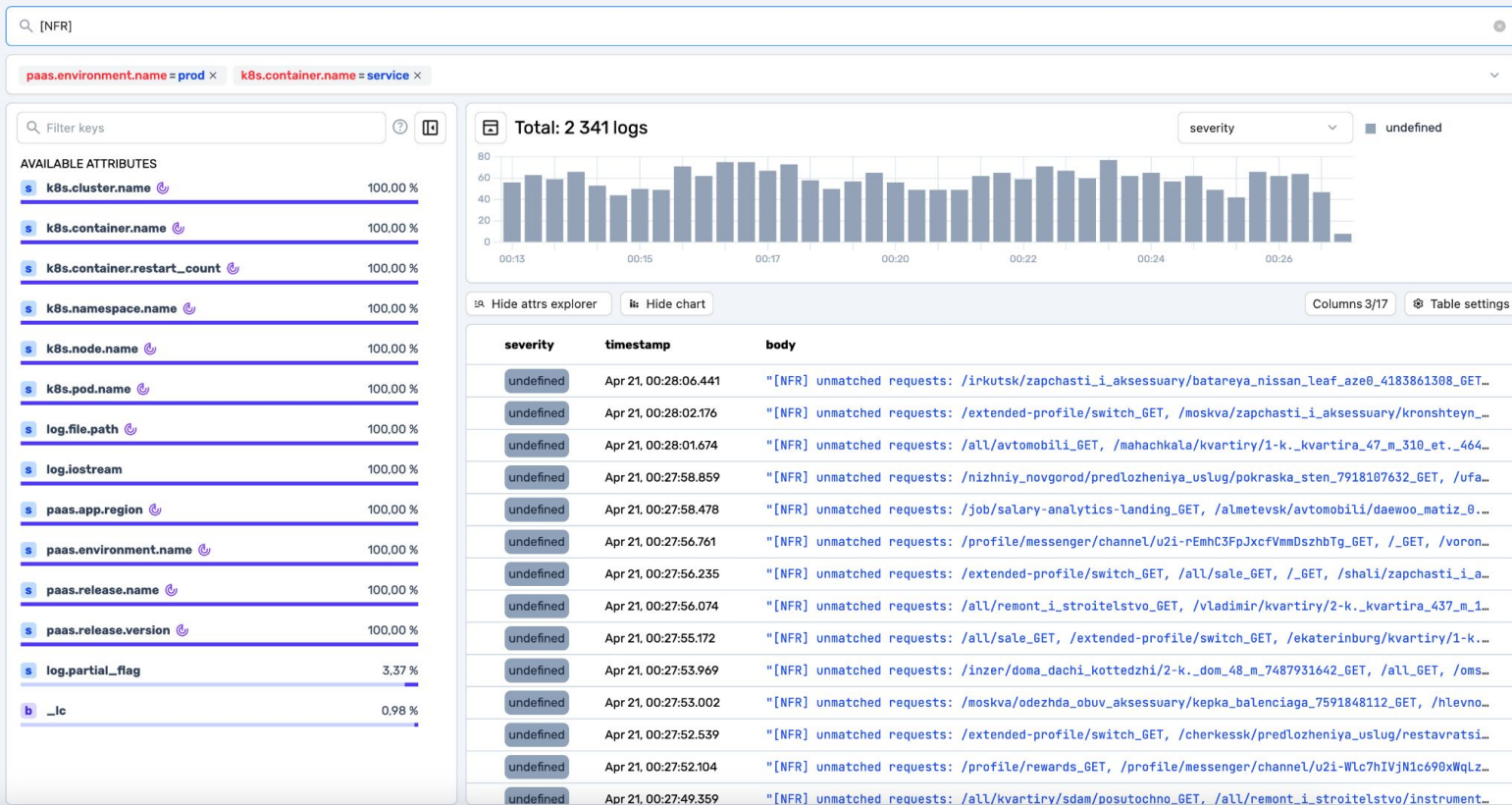
Вспомогательная информация для отладки, если требуется.

Логи





Инструменты для логов

- ✓ **OpenTelemetry** – современный открытый стандарт для сбора логов, метрик и трассировок.
-

- ✓ **ClickHouse** – масштабируемое столбцовое хранилище.
-
































Пример того, когда могут быть полезны

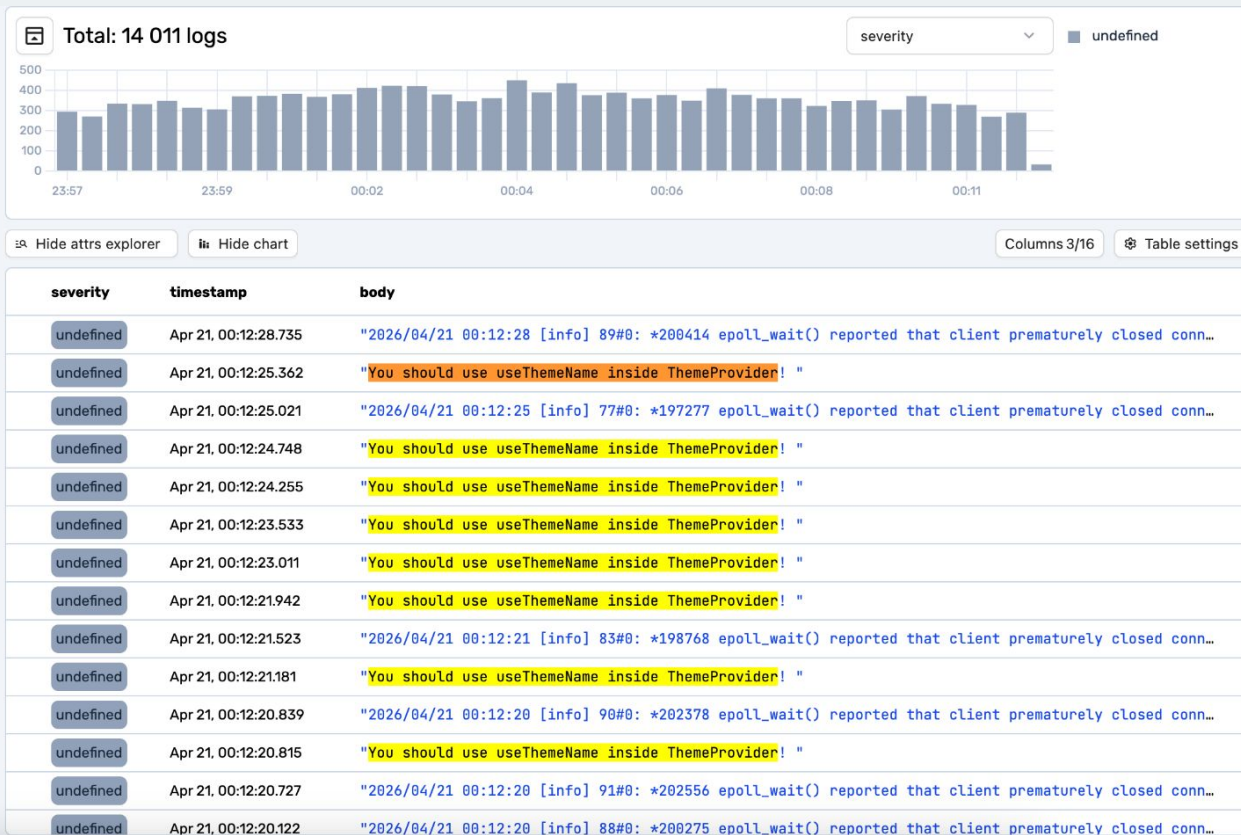
 **Log** undefined at Apr 21, 00:20:45.238   

"Store already has a same key. "

Search by keys and values

 	log.iostream	stderr
 	k8s.cluster.name	some-cluster 
 	k8s.container.name	service 
 	k8s.container.restart_count	0
 	k8s.namespace.name	some-mfe 
 	k8s.node.name	xxx 
 	k8s.pod.name	some-mfe-xxx 
 	paas.app.region	ru
 	paas.environment.name	prod
 	paas.release.name	some-mfe
 	paas.release.version	v472
 	service.name	some-mfe

Пример того, когда логи не помогают



Пример того, когда логи мешают

Логи

Инструменты для мониторинга ошибок



Sentry – платформа мониторинга производительности и отслеживания ошибок в режиме реального времени.



Sentry Microfrontends — методология настройки мониторинга ошибок, при которой микрофронтенды интегрируются с платформой Sentry.

Sentry Microfrontends

Зачем нужен инструмент:

- ⊕ нет ошибок из других сервисов;
- ⊕ в одном проекте Sentry собираем ошибки и с сервера, и с клиента;
- ⊕ команды могут настраивать свои алерты в Sentry.

My Projects ▾ | prod ▾ | 14D ▾ | 🔍 is:unresolved ✕ ⭐ 🔊

<input type="checkbox"/> ✓ Resolve ▾ 🚫 Ignore ▾ ⋮ ↕ Last Seen ▾	GRAPH: 24h 14d	EVENTS	USERS	ASSIGNEE
<input type="checkbox"/> Error _callee\$(server) toggles HTTP error! status: 503, message: upstream connect error or disconnect/re... New Issue 🔗 SERVICE-FIRST-ES 🕒 4hr ago 4d old		11	0	▾
<input type="checkbox"/> TypeError _callee\$(server) Body is unusable New Issue 🔗 SERVICE-FIRST-ER 🕒 4hr ago 4d old		11	0	▾
<input type="checkbox"/> TypeError POST /location Cannot read properties of undefined (reading 'deps') New Issue 🔗 SERVICE-FIRST-FO 🕒 16hr ago 2d old		5	0	▾
<input type="checkbox"/> SyntaxError POST /location Unterminated string in JSON at position 6 🔗 SERVICE-FIRST-DP 🕒 16hr ago 3wk old		61	0	▾
<input type="checkbox"/> Error POST /location incorrect header check New Issue 🔗 SERVICE-FIRST-EJ 🕒 1d ago 6d old		9	0	▾

Мониторинг ошибок

Пишем в ошибку всю дополнительную информацию, которая поможет при отладке.

The screenshot shows a Sentry error report for a `TypeError`. The error message is "Cannot read properties of null (reading 'getBoundingClientRect')". The report includes various filters and details:

- Filters:** browser: Yandex Browser 26.3.7, browser.name: Yandex Browser, device: V2237, device.family: V2237, environment: prod, handled: no, level: error, logger_source: browser, mechanism: instrument, os: Android 15, os.name: Android, pagelid: item-list, release: c7e8bc912911.
- Exception:** EXCEPTION (most recent call first) `App Only Full Raw`. `TypeError` Cannot read properties of null (reading 'getBoundingClientRect').
- Handled:** handled: false, function: setTimeout.
- JS Stack:**
 - JS `/mstatic/build/modern/ru-RU/pages-Main.20d0002baa123673.js` in n at line 199:599751
 - `/mstatic/build/modern/ru-RU/utilities.7a03bb3c7d3f402e.js` in b at line 1:3081
 - `/mstatic/build/modern/ru-RU/sentry.33eaf50fee62083ajs` in r at line 1:45832
- Request:** GET `/items/search` `m.avito.ru`.
- Query String:** locationId: 000.
- Headers:** Referer: `https://yandex.ru/`.
- AB:** SuperFeature:

```
{  analyticParams : [Object],  testGroup : test}
```

Мониторинг ошибок

Предотвращать проблемы

Проберы для канареечных релизов

 New Sentry errors count

Реагировать на проблемы

Алерты

 Sentry errors count spike

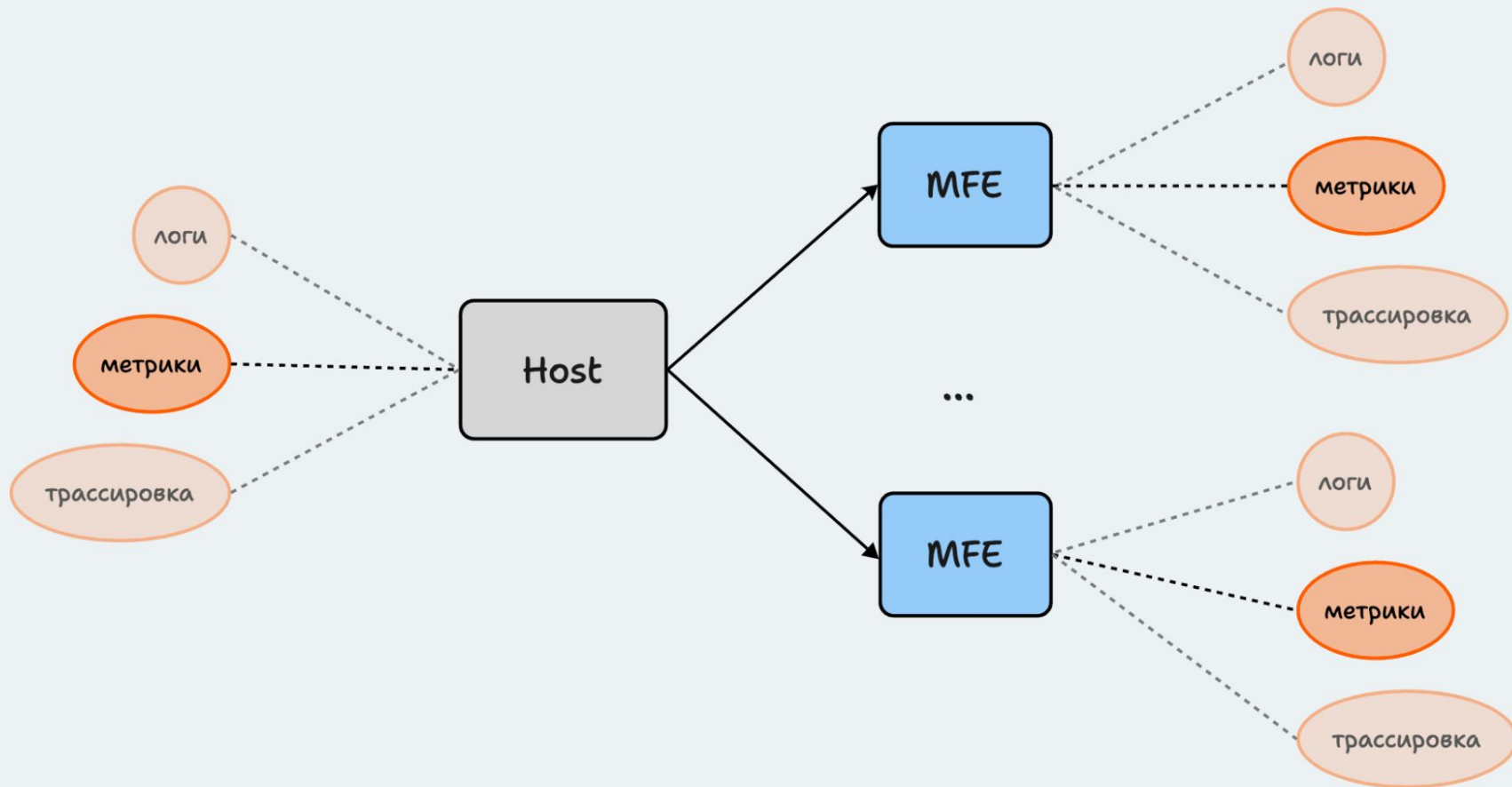
Итого. Логи

Основные принципы

- ✓ Пишем логи микрофронтенда из сервиса, в котором он находится.
- ✓ Используем инструменты, принятые в компании.
- ✓ Используем Sentry Microfrontends для роутинга ошибок.
- ✓ Исключаем ошибки, которые невозможно исправить.

Observability. Метрики

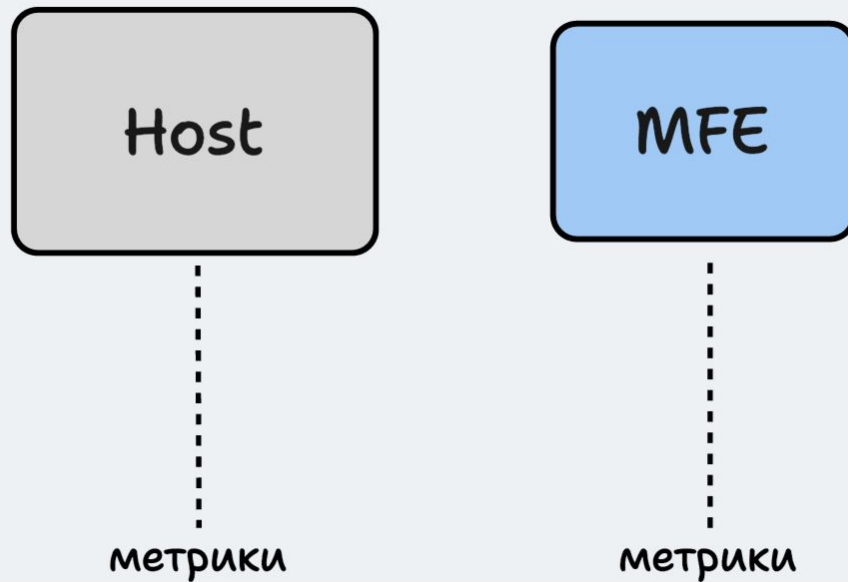




Метрики

Откуда писать

Метрики по микрофронтендам также отправляем из самого сервиса.



Метрики микрофронтенда

Что писать

- 1 **RPM (Requests Per Minute)**
- 2 **RT (Response Time)**
- 3 **HTTP Status Codes**
- 4 **API HTTP Status Codes**
- 5 **Node Metrics (CPU%, Mem%, Heap, Event Loop Delay)**

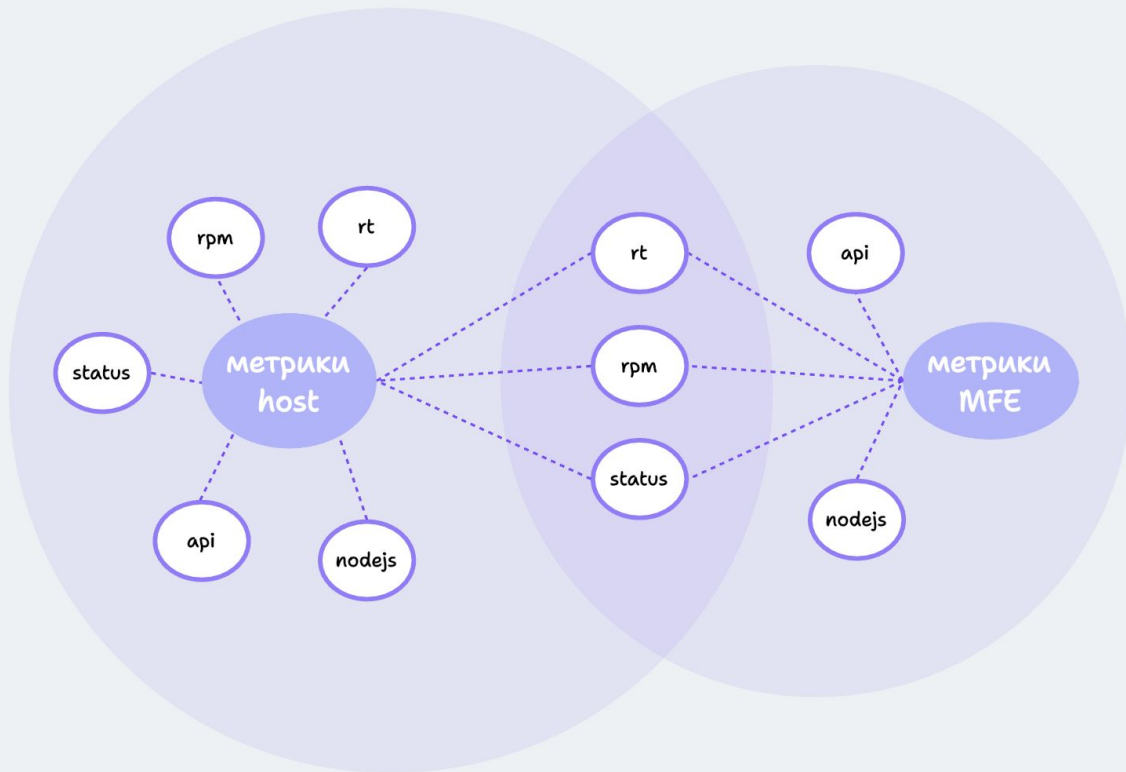


Метрики хоста

Что писать

Со стороны хоста пишем метрики микрофронтендов:

- ✓ **RPM;**
- ✓ **RT;**
- ✓ **HTTP Status Codes.**



Метрики

Как выглядит метрика

Пишем из микрофронтенда:

```
env.$env.apps.$serviceName.mfe.$mfeName.render.$sr.rt.$status
```

Обязательно
пишем environment

Имя
микрофронтенда
может быть
не уникально

SSR / CSR

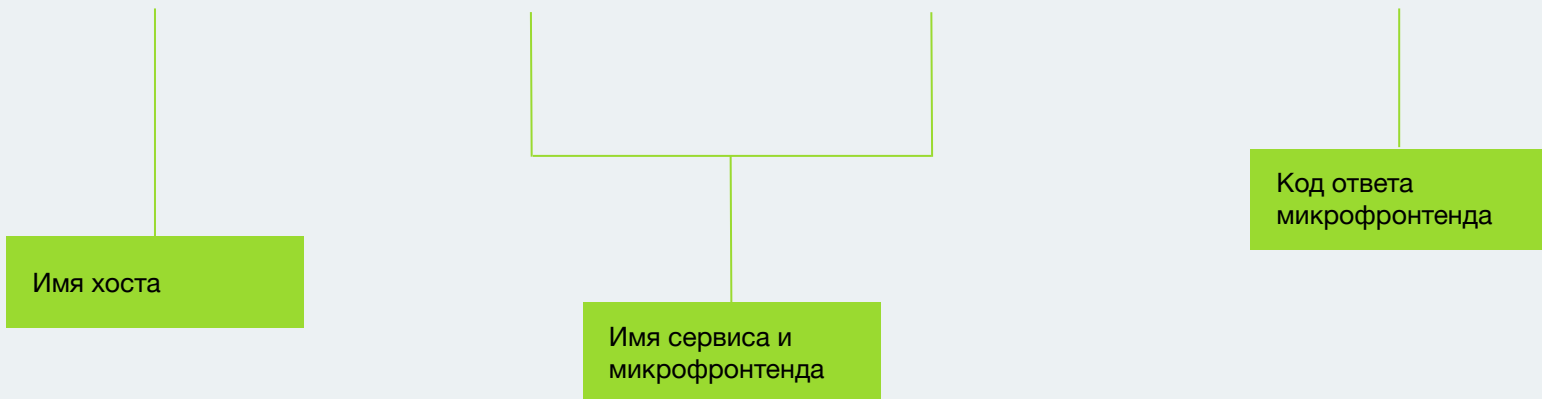
Код ответа
микрофронтенда

Метрики

Как выглядит метрика

Пишем из хоста:

```
env.$env.apps.$hostName.remote.$serviceName.mfe.$mfeName.render.$sr.rt.$status
```



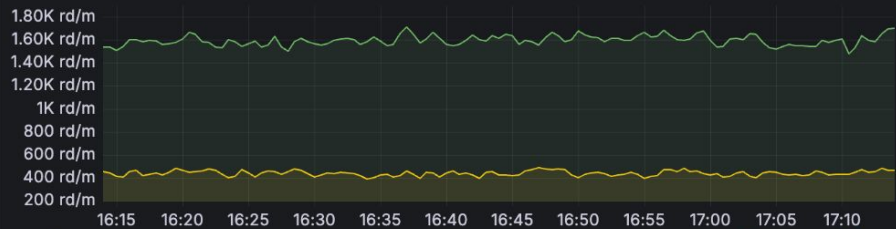
Метрики

Инструменты

- ✓ **Prometheus** — система мониторинга и оповещения для сбора, хранения и анализа метрик в реальном времени.
-

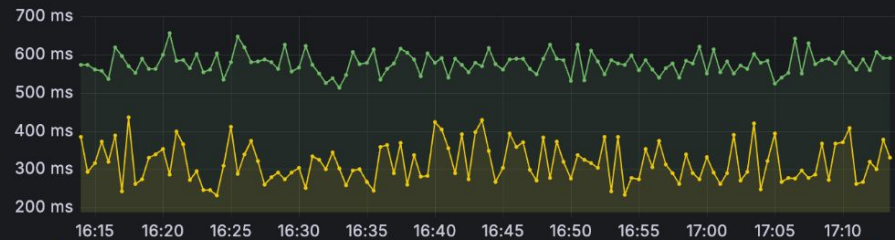
- ✓ **Grafana** — платформа для визуализации, мониторинга и анализа данных в реальном времени.
-

requests per minute



Name	Mean	Max
foo.200	1.59K rd/m	1.71K rd/m
bar.200	437 rd/m	486 rd/m

response time [%%98]



Name	Mean	Max
foo.200	578 ms	657 ms
bar.200	318 ms	437 ms

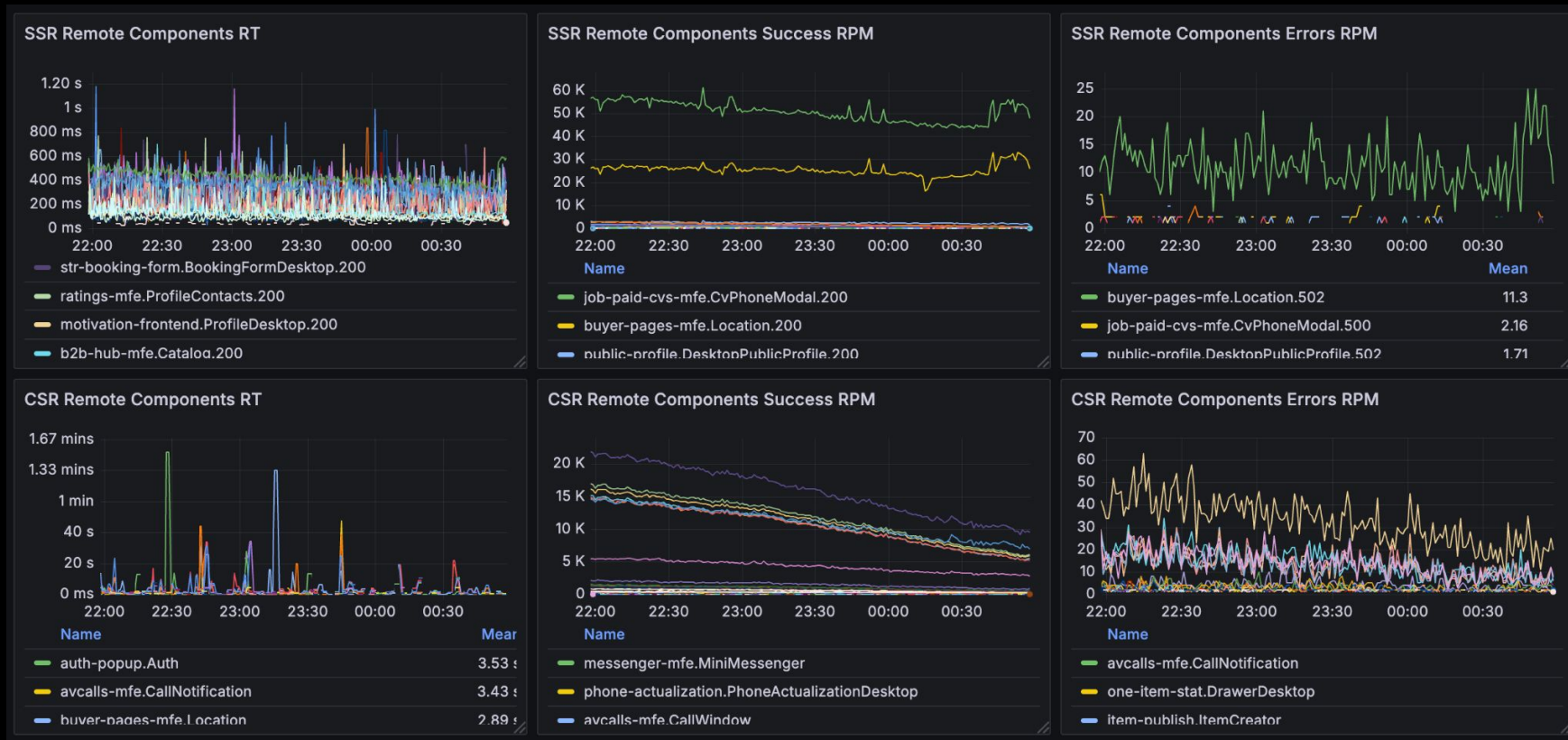
4XX ERRORS

No data

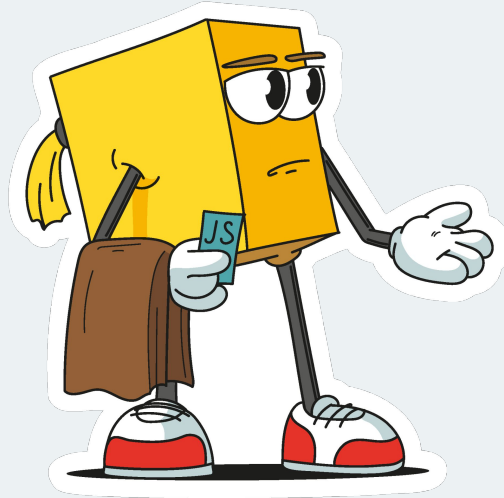
5XX ERRORS

No data

Результат вывода метрик со стороны сервиса с микрофронтендами



Результат вывода метрик со стороны хоста



**Например, спросить владельцев,
почему так**

Метрики

Применение

Предотвращать проблемы

Проберы для канареечных релизов

● 5xx spike

● RT spike

● Mem/CPU climb

Реагировать на проблемы

Алерты

● 5xx spike

● RT spike

● Mem/CPU climb

Измерять успех

NFR

● 5xx count

● RT

**Самое сложное –
настроить лимиты
для пробера, чтобы
исключить false positive**



Лимиты для проберов

● 5xx spike

● New Sentry errors count

Можем выставить статическое значение, так как по ним редко случаются выбросы.

● RT spike

● Mem/CPU climb

Будет отличаться в зависимости от времени суток, дня недели и так далее, поэтому статическое значение не подойдет.

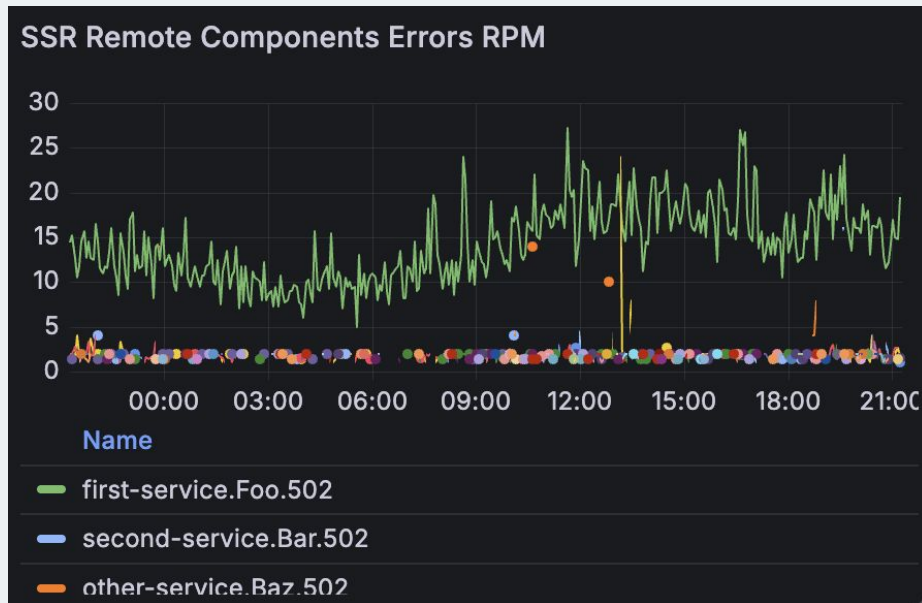
Лимиты для проберов

Пример

● 5xx spike

```
// значение для остановки канарейки  
stop: ">50"
```

```
// значение для отката канарейки  
rollback: ">100"
```

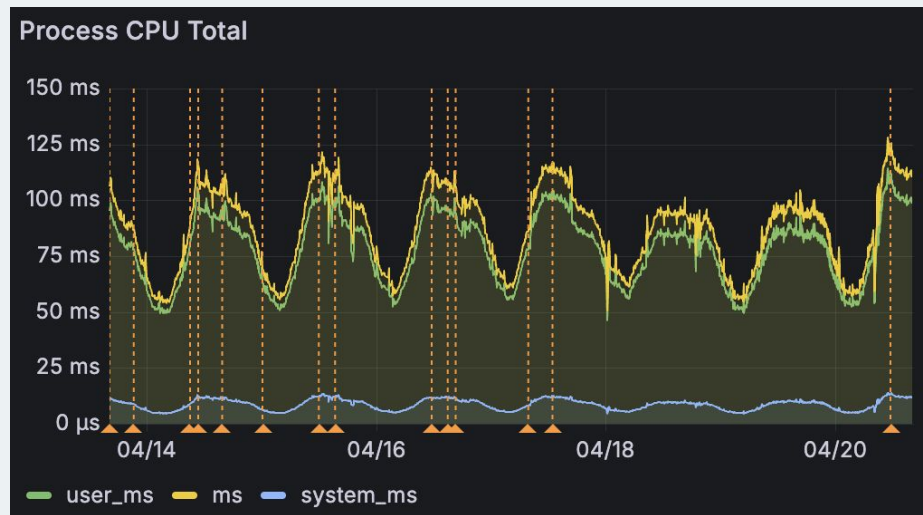


Лимиты для проберов

Пример

● CPU climb

Здесь подойдет
baseline
за неделю



Лимиты для проберов

Пример

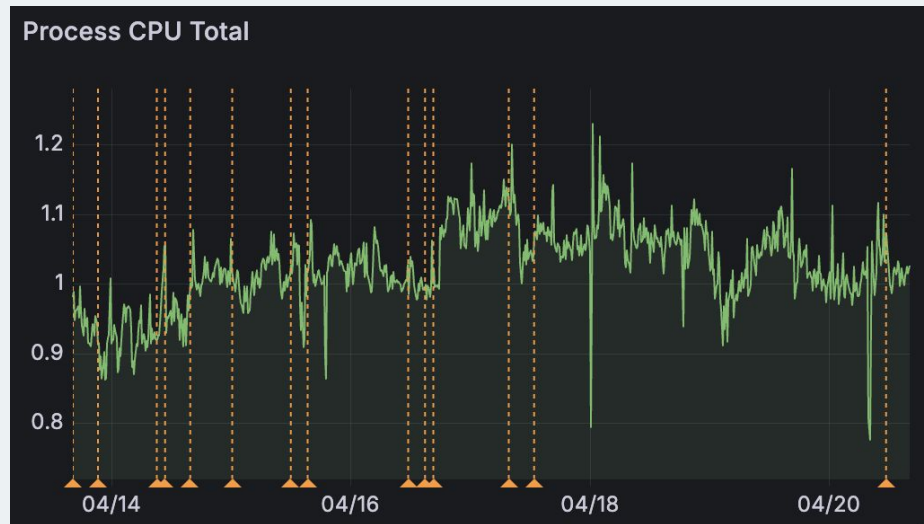
● CPU climb

```
// значение для остановки канарейки
```

```
stop: ">1.2"
```

```
// значение для отката канарейки
```

```
rollback: ">1.5"
```



NFR

Пример

- RT
- 5xx count

```
// должен отвечать за 100ms  
latency = "100ms"
```

```
// ожидаемая нагрузка  
rpm = 100
```



Метрики

Пример

Что еще можно писать со стороны хоста и микрофронтендов:

- перформанс метрики;
- продуктовые метрики;
- события event-bus.

Метрики



Высокая кардинальность – характеристика набора данных, указывающая на то, что поле содержит большое количество уникальных значений.

Высокая кардинальность метрики

Пример

Допустим, мы ходим за данными в ручку
`/user/{userId}`.

Высокая кардинальность метрики

Пример

Тогда метрики могут выглядеть так

```
`api./user/123.rt.${status}`  
`api./user/456.rt.${status}`  
`api./user/789.rt.${status}`
```

Плохо, слишком
много значений

Высокая кардинальность метрики

Пример

Правильнее сделать так

```
`api.getUser.rт. ${status}`
```

Так как это на самом деле одна ручка

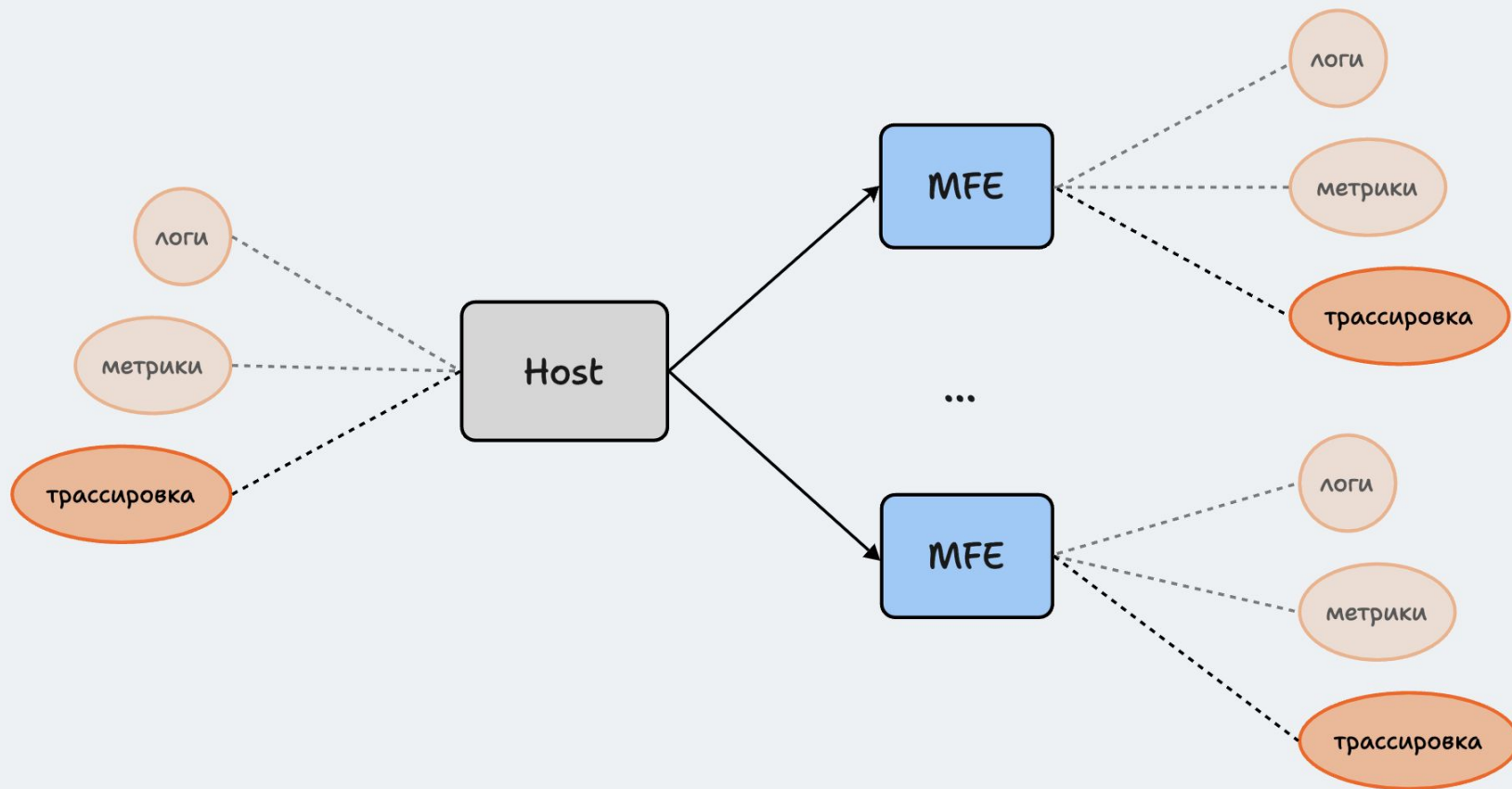
Итого. Метрики

Основные принципы

- ✓ Покрываем метриками с обеих сторон.
- ✓ Разделяем сервер и клиент.
- ✓ Избегаем метрик с высокой кардинальностью.

Observability. Трассировка

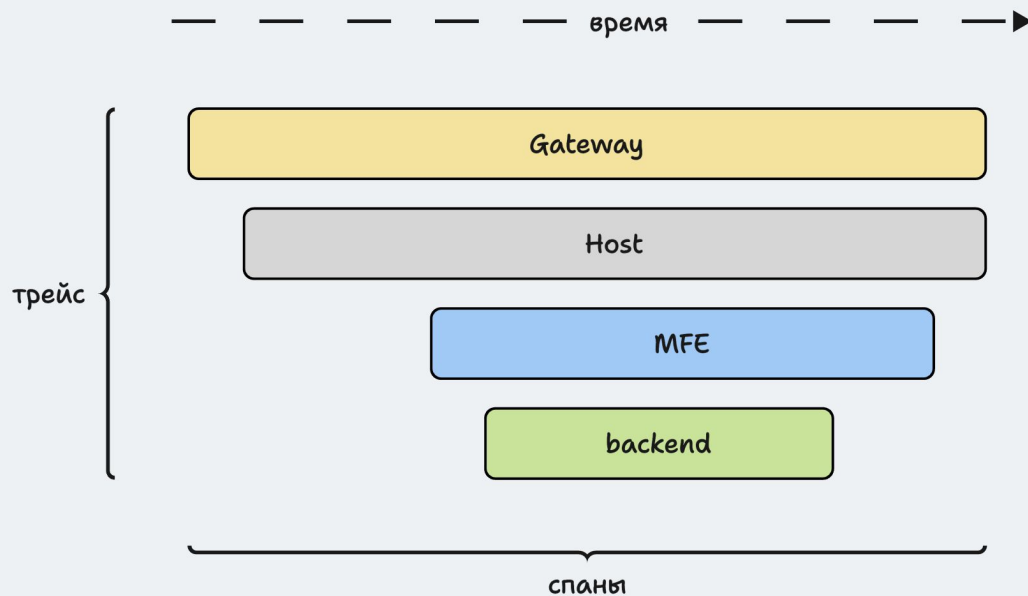




Трассировка

Как связываются запросы

Каждый сервис добавляет свой «**спан**» с меткой времени, статусом и контекстом.



Трассировка

Инструменты

✓ **OpenTelemetry Collector** собирает трейсы, агрегирует их и отправляет в централизованную систему хранения.

✓ **Jaeger** используется в качестве бэкенда для хранения и визуализации трассировок.

all staging prod All ⊖ Errors

Jaeger-debug-id Select

Service

Operation Select

Dur. Not specified

Period now-15m

Tags Not specified

Find traces ?

personal-items / egress 1.09ms

3 spans start: 22:36:41.027
end: 22:36:41.028

personal-items (1) toggles (2) 970f9a75d2c97a81197ee76c788c8ee9

istio-ingressgateway / egress 45ms

12 spans start: 22:36:34.429
end: 22:36:34.474

avito-api-gateway (9) istio-ingressgateway (1) personal-items (2) 261717ff3bf106a5dc0e3ca4ff265be5

istio-ingressgateway / egress 412ms

⊖ 2 errors 880 spans start: 22:36:33.922
end: 22:36:34.333

- abac (38) accounts-hierarchy-api (9) accounts-hierarchy-employees (12) adv-agency (21) auto-vas (2)
- autoload-customers (15) avatar (12) avito-api-gateway (76) avito-desktop-site (8) billing-avito-gateway (6) breadcrumbs (5)
- bundle (2) comtrans-dealers (4) contractor (6) delivery-item (2) draft-storage (4) error-observability (1) favorite-sellers (3)
- favorites (3) geo-resolver (9) geo-storage (3) georgy (6) image-storage (12) infomodel-classificator (4)
- infomodel-validation (30) istio-ingressgateway (1) item (28) item-autopublish2 (3) item-platform (53) item-verification (4)
- job-chat-bots-mfe (2) job-employer (4) job-employer-cv (6) laas-frontend (3) layout (49) listing-fees-storage (6)
- marketplace-sale (4) messenger-integration-api (4) messenger-storage (5) mnz-bbl (3) mnz-freemium (4)
- mnz-item-liquidity-limits (4) mnz-sf (5) mobile-api-common (10) mp-abuse (2) mrt-entry-points (8) mrt-inventory (4)
- nonliquid-storage (4) padme (5) personal-items (4) pets-benefits (3) portfolio (5) pricing-discounts (2) pro-access (3)
- pro-entry-points (9) pro-search (3) profile-type (6) ratings (4) ratings-state (5) re-mortgage-entrypoint (8) realty-users (2)
- referral-platform-backend (4) reputation-gateway (4) request-scoring (6) reservation (3) rfp (5) seller-audience (5)
- seller-dashboard-core (10) seller-fill-parameters (4) seller-item-tips (3) serp (60) session (18) soa-stats (4)
- split-tests-go (48) statistics-facade (4) statistics-fast-storage (4) subscription-sx-services (5) tariff-aggregator (30)
- tariff-features (24) u8r (6) urgent (3) user-limits (5) user-profile (30) user-toggles (27) 23343a48c5aa60e6bf4d4f854305798f

Пример визуализации трассировок

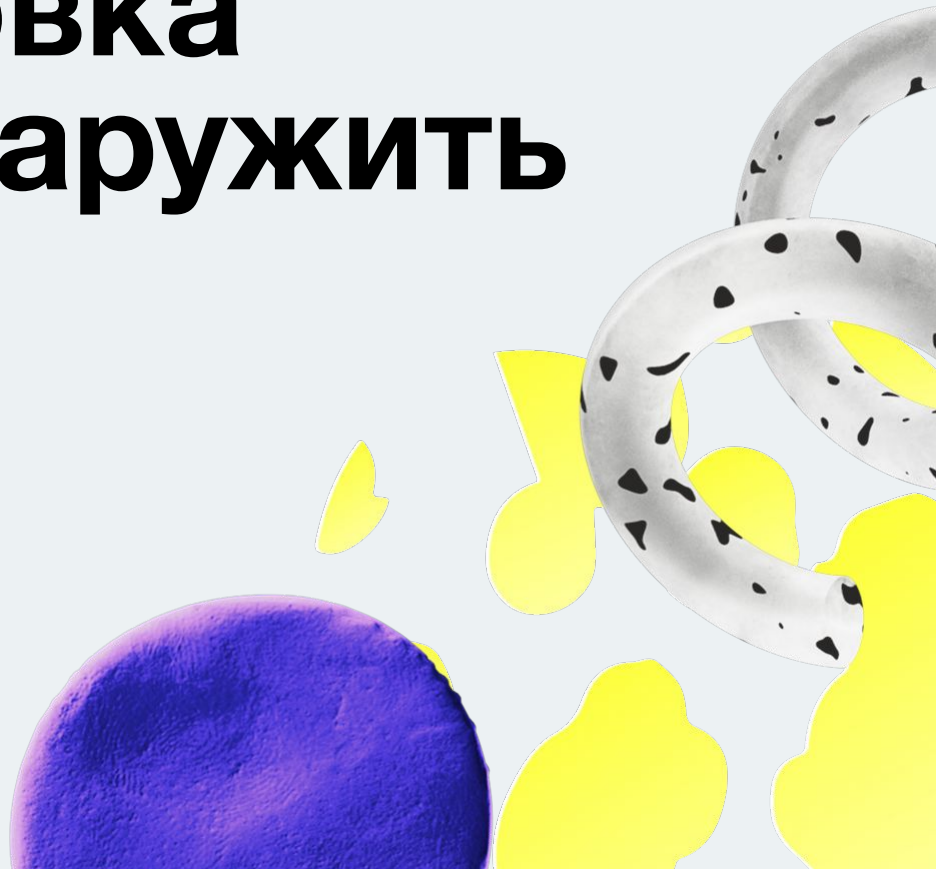
Трассировка

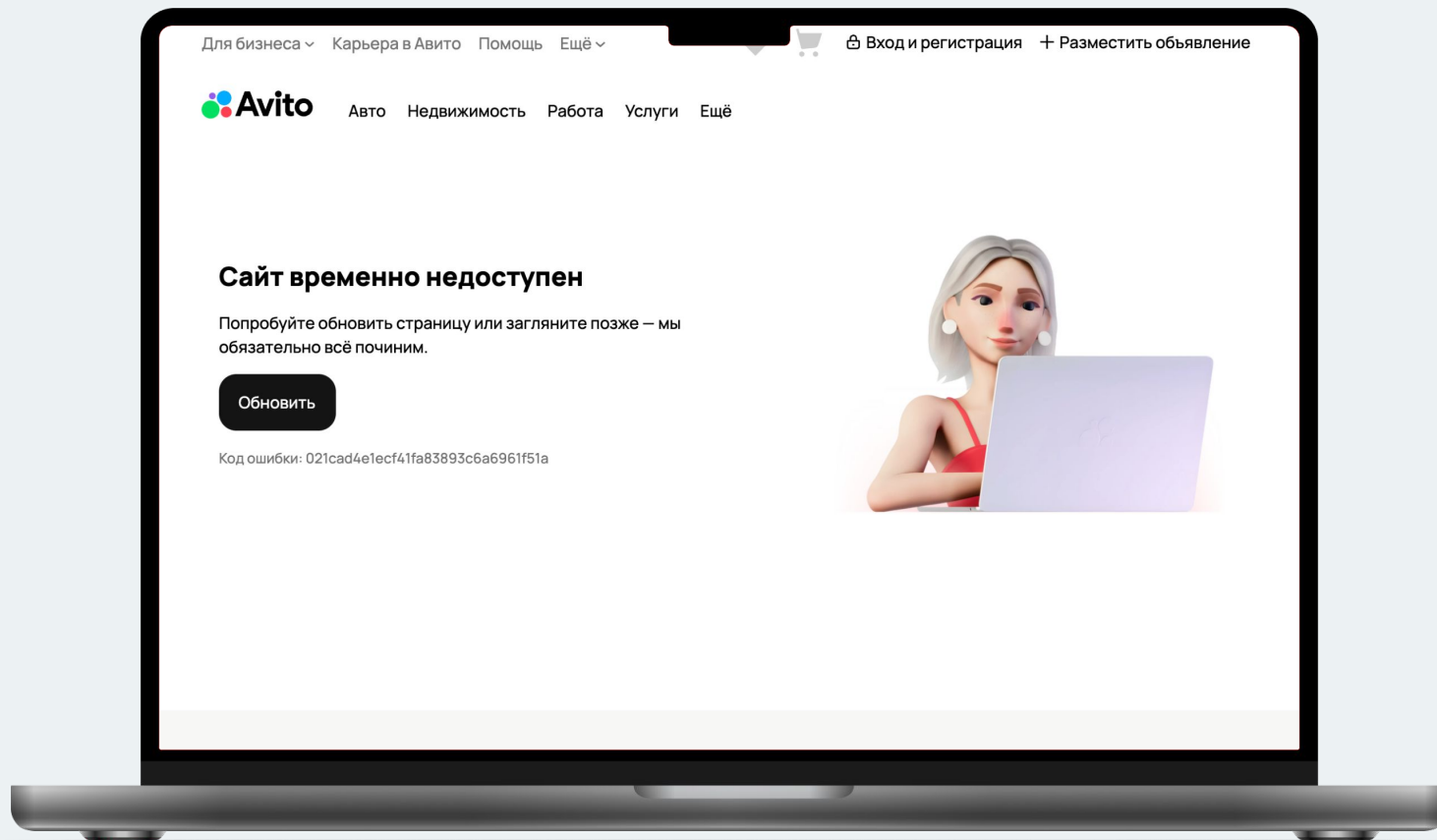
Пример:

spans 12 services 3 depth 9 duration 45ms started at 22:36 (10 minutes ago)



Как трассировка помогает обнаружить проблему

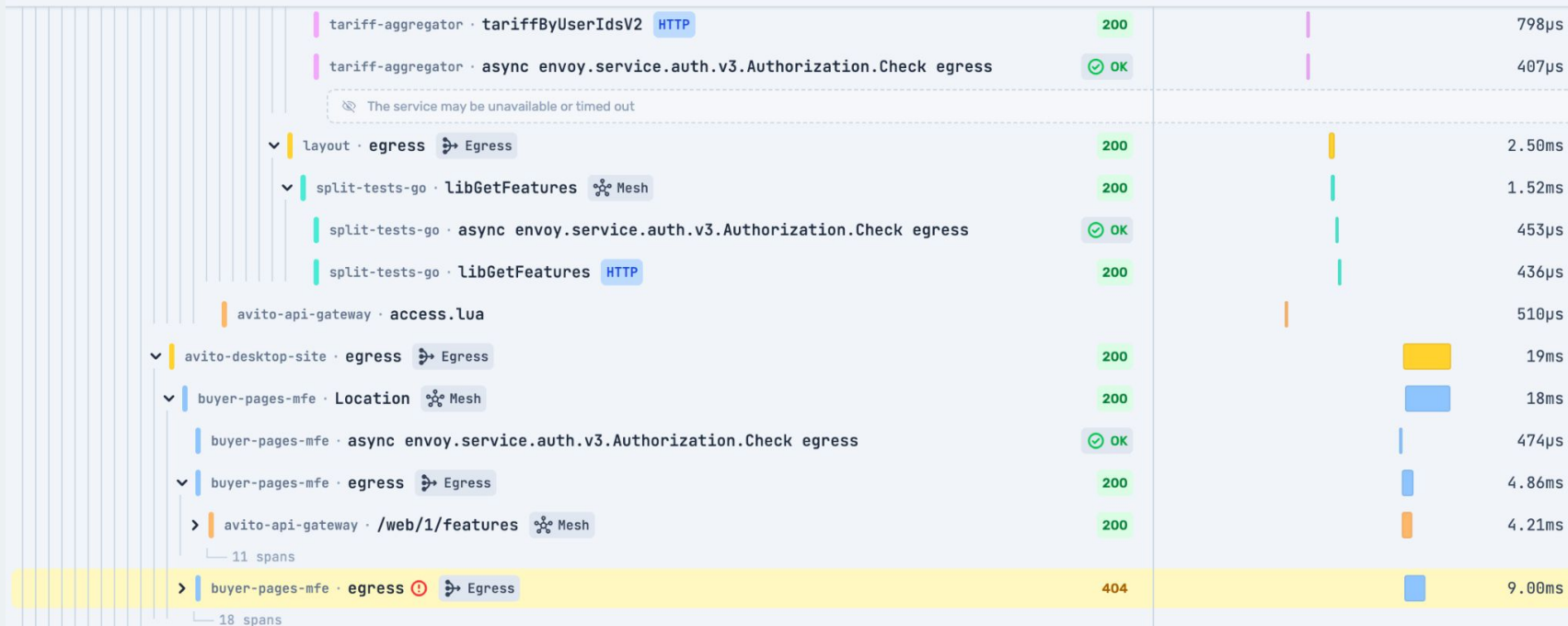




Трассировка

Пример:

spans 145 services 13 depth 26 duration 136ms started at 22:57 (5 minutes ago)



Трассировка

Пример:

spans 145 services 13 depth 26 duration 136ms started at 22:57 (5 minutes ago)

Service	Operation	Status	Duration
avito-desktop-site · egress	Egress	400	18ms
avito-api-gateway · /web/1/getBanner	Mesh	400	9.46ms
avito-api-gateway · firewall	HTTP	400	9.89ms
avito-api-gateway · /	HTTP		3µs
avito-api-gateway · @default	HTTP	400	9.15ms
avito-api-gateway · access.Lua			372µs
avito-api-gateway · /web/1/getBanner	HTTP	400	7.33ms
avito-api-gateway · /web/1/getBanner	HTTP	400	7.09ms
avito-api-gateway · mapi-init.lua			268µs
avito-api-gateway · egress	Egress	400	6.33ms
bx-api · /web/1/topBanner	Mesh	400	5.61ms
bx-api · async envoy.service.auth.v3.Authorization.Check egress		OK	750µs
bx-api · /web/1/getBanner	HTTP	400	4.15ms
bx-api · egress	Egress	200	3.31ms

Span id: 0844ac6a38de16cc service: bx-api operation: /web/1/topBanner duration: 4.15ms

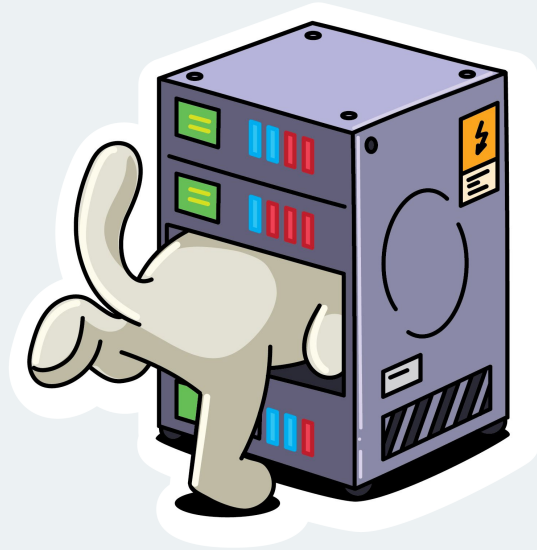
Tags Process Logs (1)

timestamp	Критичность	Тело лога
Apr 19, 22:57:16.912	warn	{cluster_name:"xxx",env:"prod",error:"empty response for build searchForm by url",function:"go.avito.ru/bx/service-bx-api/internal/api/handler/web_get_banner.(Handler).Handle",k8s_node:"xxx",k8s_pod:"bx-api-xx",lineno:64,message:"",tag:"service.bx-api",time:"2026-04-19T19:57:16Z"}
		<ul style="list-style-type: none"> cluster_name: xxx env: prod error: empty response for build searchForm by url k8s.container.name: service k8s.container.restart_count: 0 k8s.namespace.name: bx-api

Ошибка на
бэкенде

Итого. Трассировка

- ✓ Помогает в отладке микрофронтендов.
- ✓ Позволяет обнаружить долгие ответы.



Главное – всё это мы пишем «из коробки»!

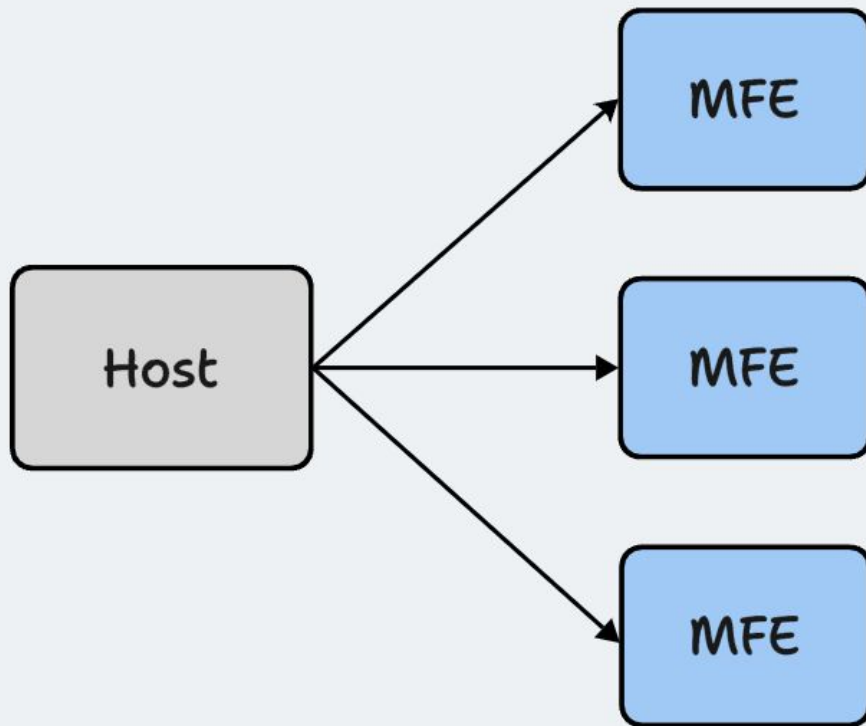
Разберём, как мы добились такого результата



Реализация платформенного решения

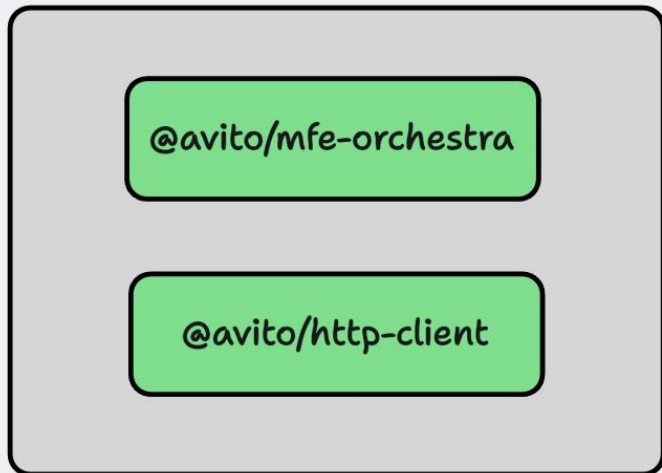


Архитектура микрофронтендов



Как это выглядит на уровне ниже

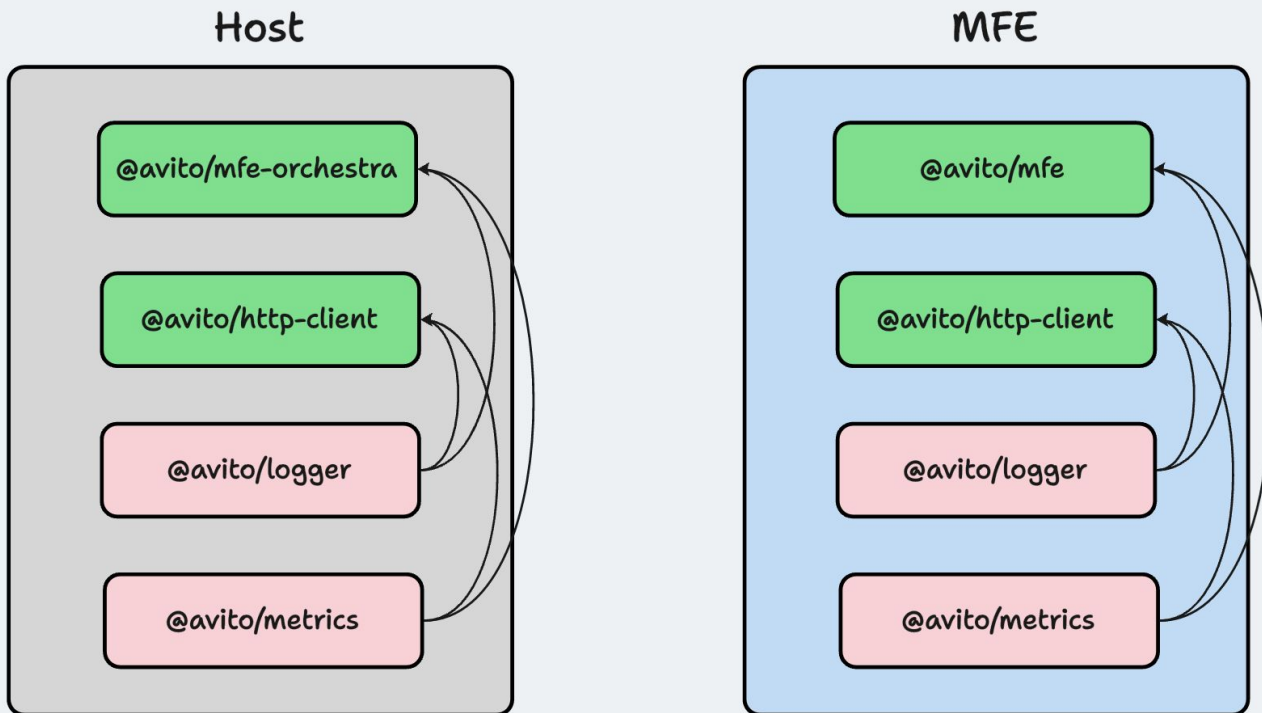
Host



MFE



Решение для логов и метрик



Решение для логов и метрик

```
env.$env.apps.$serviceName.mfe.$mfeName.render.$sr.rt.$status
```



@avito/mfe

```
env.$env.apps.$hostName.remote.$serviceName.mfe.$mfeName.render.$sr.rt.$status
```



@avito/mfe-orchestra

```
env.$env.apps.$serviceName.mfe.$mfeName.api.$sr.rt.$status
```



@avito/http-client

Решение для логов и метрик

Реализуем платформенные библиотеки **logger** и **metrics**

Переменные
подставляются
во время деплоя

```
// @avito/logger
Logger.init({
  sentryDsn: process.env.SENTRY_DSN,
  environment: process.env.APP_ENV,
  release: process.env.APP_RELEASE
});
```

```
// @avito/metrics
Metrics.init({
  prefix: process.env.APP_NAME
});
```

Создание сервиса

avito service create



Микрофронтенды > SuperFeature

SuperFeature

 Grafana  Logs  Stash  Sentry

Основное

О микрофронтенде

Сервис

 new-functionality

Описание

Сервис с новой бизнес-функциональностью

Добавление базовых метрик

Через Grafana library panels

01 Упростить старт

Платформенные метрики не нужно добавлять вручную – они будут доступны сразу.

02 Доставлять обновления

Путей метрик во все микрофронтенды при необходимости.

env prod percentile 98 serviceName signal percentile.75 interval 5m Last 6 hours Refresh

> SLA (2 panels)

main

requests per minute	response time [%98]	5XX ERRORS	4XX ERRORS
No data	No data	No data	No data

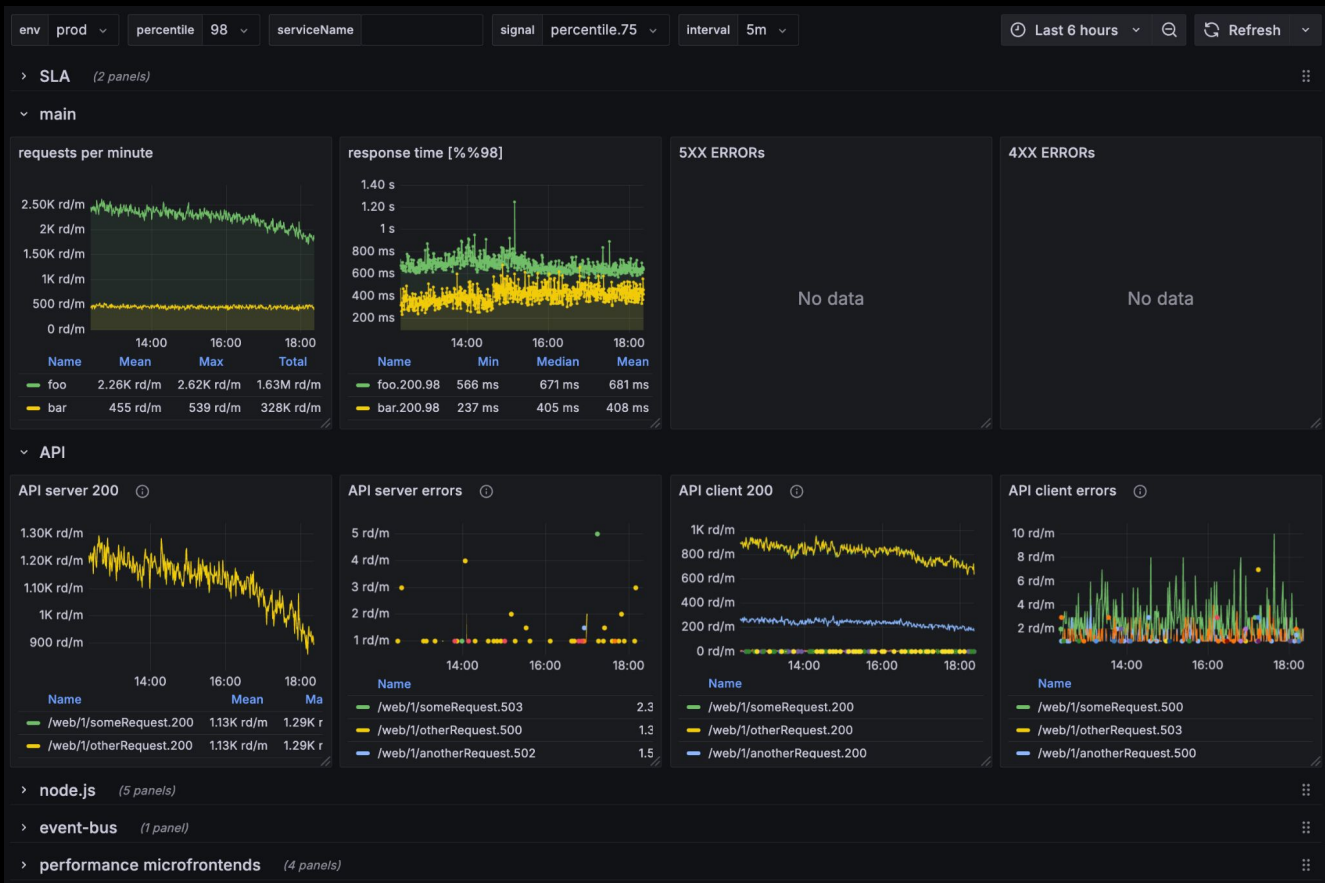
API

API server 200	API server errors	API client 200	API client errors
No data	No data	No data	No data

> node.js (5 panels)

> event-bus (1 panel)

> performance microfrontends (4 panels)



Платформенное решение позволило

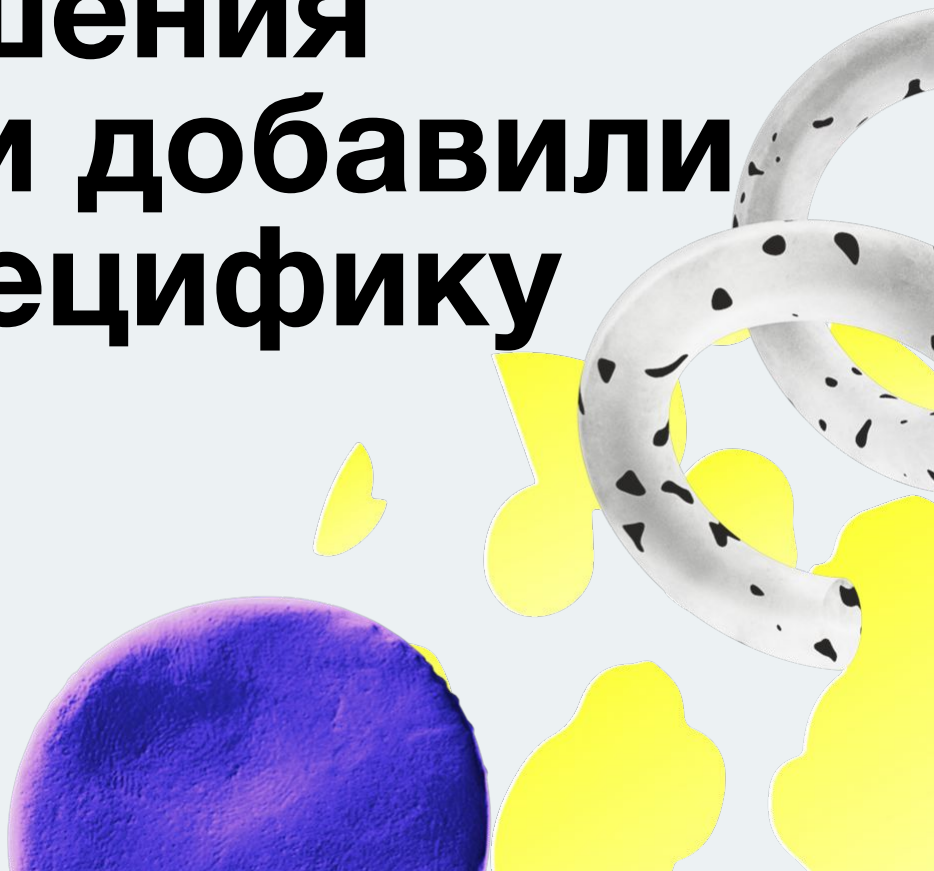
- ✓ **Упростить настройку observability с нуля**
благодаря тому, что основное пишем «из коробки».
- ✓ **Упростить погружение в чужой проект**
благодаря унифицированности.
- ✓ **Нивелировать риск забыть об observability**
благодаря тому, что в базовом виде оно всегда есть.

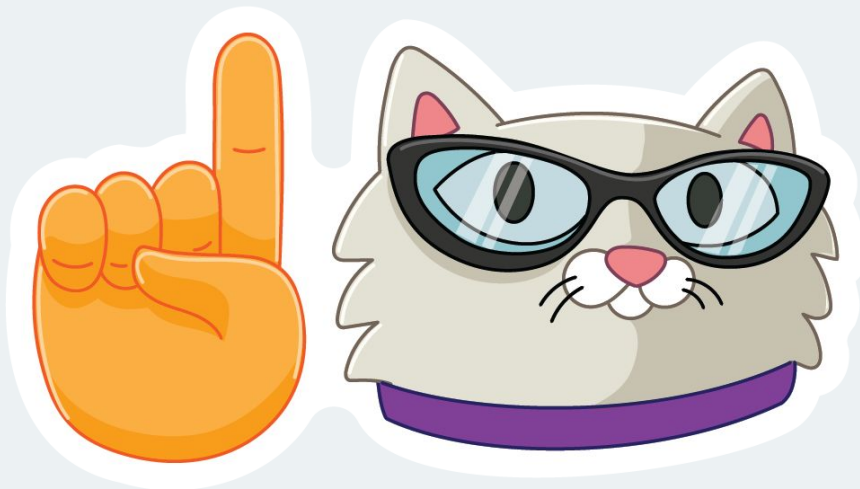


ИТОГИ



Мы взяли решения для бэкенда и добавили фронтенд-специфику





**Невозможно предотвратить все ошибки.
Но можно **построить систему**, которая **будет
работать** тогда, когда ошибаются люди**

Контакты

Напишите, если у вас
остались вопросы:



Дарья Саенко
Frontend Engineer

✉ dariassaenko@gmail.com